SCC-R-102-3

OBJECT DISCRIMINATION VIA BULK TEMPERATURE/VELOCITY FILTERING

Small Business Innovation Research Topic No. 87-3

PHASE I FINAL REPORT

Contract No.N00014-87-C-0801

12 April 1988

William B. Kendall, Principal Investigator

Submitted to:

Scientific Officer
Mathematical and Physical Sciences Directorate
Mathematics Division
Office of Naval Research
Attn: Dr. Keith Bromley (Code 1111MA)
Ref: Contract N00014-87-C-0801
800 N. Quincy Street
Arlington, Virginia 22217

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY —	3. DISTRIBUTION/AVAILABILITY OF REPORT
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE —	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) SCC-R-102-3	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION SPACE COMPUTER CORPORATION 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION Office of Naval Research
6c. ADDRESS (City, State, and ZIP Code) 2800 Olympic Blvd., Suite 104 Santa Monica, CA 90404-4119	7b. ADDRESS (City, State, and ZIP Code) 800 N. Quincy Street Arlington, VA 22217
8a. NAME OF FUNDING/SPONSORING ORGANIZATION --- 8b. OFFICE SYMBOL (If applicable) ---	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER Contract No. N00014-87-C-0801 Solicitation No. SBIR 87.1
8c. ADDRESS (City, State, and ZIP Code) ---	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)

OBJECT DISCRIMINATION VIA BULK TEMPERATURE/VELOCITY FILTERING

12. PERSONAL AUTHOR(S)

KENDALL, William B.

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Final Technical	FROM 87Aug14 TO 88 Feb14	88APR12	63

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	
			Discrimination; Signal Processing; Computer Architecture; Bulk Filtering; Image Processing; Parallel Processing; Associative Processing; Algorithms

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

This report describes work on the velocity filter approach to target detection and tracking, including its application to kinetic kill debris rejection. We present results of theoretical studies deriving the optimum (maximum-likelihood) filter for detecting a target moving through a sequence of N passive sensor image frames with a known, assumed or estimated vector velocity in which the target and background observations are modeled as independent Poisson processes and where both uniform and nonhomogeneous backgrounds are considered. Also presented are the results of algorithm simulation experiments conducted with (a) digital imagery recorded by the ~~MIT~~ Lincoln Laboratory 30-inch "Quad-Camera" optical telescope and (b) simulated target imagery. Finally, a study of alternative approaches to processor implementation is presented. The implementation study includes consideration of a fine-grained, massively-parallel associative processor.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL Dr. Keith Bromley (Code 1111 MA)	22b. TELEPHONE (Include Area Code) (619) 553-2535 / 22c. OFFICE SYMBOL

DD Form 1473, JUN 86 *Previous editions are obsolete.* SECURITY CLASSIFICATION OF THIS PAGE

UNCLASSIFIED

TABLE OF CONTENTS

1. INTRODUCTION

This report describes work carried out by Space Computer Corporation under a Small Business Innovation Research (SBIR) Phase I program sponsored by the Innovative Science and Technology Office of the Strategic Defense Initiative Organization (SDIO/IST) and managed by the Office of Naval Research under Contract No. N00014-87-C-0801. The original purpose of this Phase I effort was to study object discrimination via bulk temperature/velocity filtering, including the following specific objectives:

a. Develop basic algorithms and evaluate system performance with clusters of moving objects of different types with various forms and levels of background clutter and noise;

b. Develop basic processor architecture, including specification of processor operations and throughputs, memory sizes and read/write delays, bus data rates, etc.;

c. Evaluate feasibility of processor implementation with low-cost commercial components for Phase II brassboard system.

We in fact enlarged the area of study to include object detection and tracking as well as discrimination for the reasons mentioned below.

In current approaches to passive IR sensor signal and data processing for surveillance and fire control systems, object discrimination is performed with track-file data after scan-to-scan correlation. The scan-to-scan correlation itself is performed with classical "track-while-scan" track-association techniques. However, these approaches can encounter major difficulties when large numbers of objects such as booster fragments and kinetic kill debris are present in the field-of-view. In the first place, evidence is accumulating hat such fragments and debris may have very rapid fluctuations in intensity due to tumbling and other motions, with frequency components up to tens of Hertz. In order to distinguish these fragments and debris from RVs, it is desirable to observe and process data at an appropriately high sampling rate. Since the effective sampling rate after scan-to-scan correlation is very low, e.g. 0.05-0.20 Hz for typical scan periods of 5-20 seconds, object discrimination based upon track-file data only will not contain all of the information which could be used for discrimination.

The second difficulty results from the sensitivity of classical track-association techniques to such conditions as merging/crossing tracks,

temporary loss of input data, poor background rejection, etc. The resulting track mis-associations cause errors in the intensity-vs.-time profiles of tracked objects; these result in additional discrimination errors.

The first difficulty may be alleviated by performing object discrimination in two stages. The first stage is carried out early in the processing chain using data directly from the focal plane array where the individual detectors can provide a high sampling rate. With this approach, we utilize the multiple TDI columns for observation of high-frequency intensity fluctuations (for strong targets which can be detected without TDI) for the first stage of object discrimination. This will allow high-frequency components to be detected and analyzed to determine both spectral (temperature) and temporal (fluctuation) characteristics. This first stage of object discrimination might also provide significant background suppression (e.g., rejection of hot stars) and reduction of the total number of objects to be processed later.

The second difficulty with existing approaches, that of track mis-association, can be alleviated by the use of velocity filtering techniques. These techniques provide computationally-efficient track association combined with signal-to-noise ratio enhancement. The algorithms are extremely robust with respect to such conditions as merging/crossing tracks, loss of input data, etc., even with very large numbers of objects, and should reduce second-stage discrimination errors due to track mis-association to a lower level. Velocity filtering also provides additional background suppression and may permit automatic rejection of debris resulting from kinetic kills.

The second stage of object discrimination is performed after velocity filtering and tracking to detect "leakers" from the previous processing steps as well as decoys and other penetration aids with more slowly-fluctuating intensity profiles.

Figure 1 shows the algorithm processing chain for the new approach. It should be noted that the first stage of object discrimination and TDI are functionally integrated to maximize the use of available information from the focal plane array.

During Phase I we developed algorithms for object discrimination, background suppression, track association and kinetic kill debris rejection based upon the above concepts. We also evaluated alternative processor architectures capable of executing these and other required algorithms.

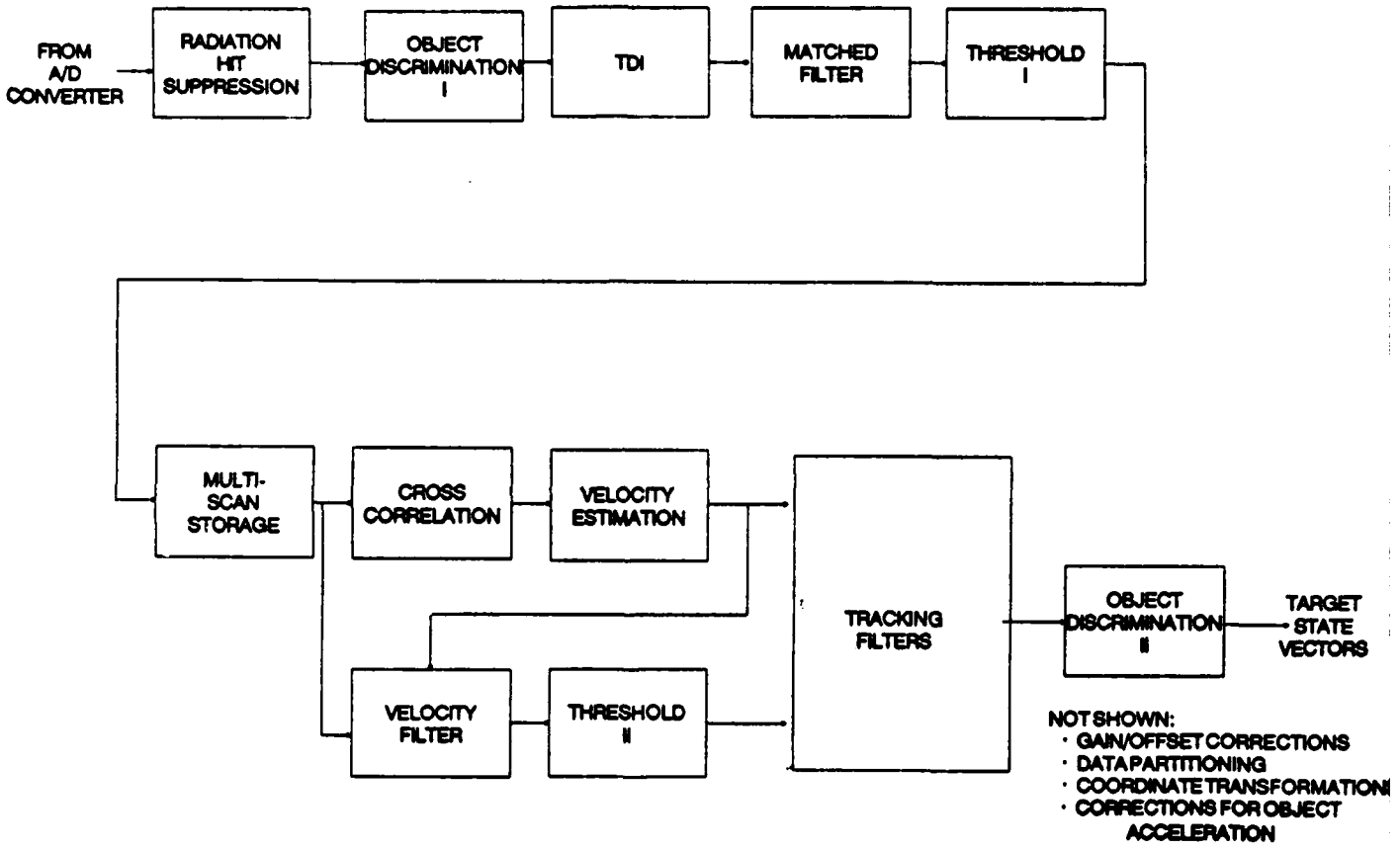

Figure 1. Algorithm Processing Chain.

Architectures investigated included a wide variety of parallel machines, both special- and general-purpose in nature, and encompassed both the time-dependent and object-dependent portions of the overall processing chain. We concluded that a programmable, fine-grained parallel processor architecture was the most attractive from the viewpoints of flexibility, long-term reliability and performance.

Of the various fine-grained architectures we investigated, the Associative String Processor (ASP) appears to be the most attractive. The ASP is a fully programmable, fine-grained parallel processor developed by Brunel University and Aspex Microsystems Ltd. in the U.K. The ASP operates in conjunction with a conventional general-purpose host computer to provide extremely high throughput. It is ideal for algorithm execution because its architecture matches closely the highly parallel algorithms and data structures utilized. It is also well suited for flight model implementation; our studies show that a fault-tolerant, radiation-hard ASP machine with over 64,000 processing elements and a throughput of 20K - 640K MOPS can be packaged in a four-inch cube using currently-available hybrid or monolithic wafer-scale integration technology.

2. DESCRIPTION OF THE WORK CARRIED OUT

The basic objectives of Phase I as set forth in our original proposal were to perform (a) proof-of-concept computer simulations and (b) processor feasibility studies in preparation for a brassboard demonstration program with simulated and/or actual sensor inputs during Phase II. Specific technical objectives included the development of basic algorithms and evaluation of system performance with background clutter and noise, as well as the development of basic processor architecture and the evaluation of processor objectives. Our Phase I work in these areas is summarized in the sections which follow. It should be noted that our ideas on combined temperature-velocity filtering have been modified somewhat from those presented in our original proposal. This is a result of new evidence concerning the possibility that booster fragments and kinetic kill debris may have very rapid fluctuations in intensity, and that object discrimination performed only after scan-to-scan correlation may therefore not be sufficient.

2.1 Velocity Filter Approach to Target Detection and Tracking

2.1.1 General. The velocity filter algorithm performs the functions of target acquisition, track initiation, and scan-to-scan correlation for input time sequences of passive IR images. It offers a number of performance advantages which can be used to augment traditional tracking approaches, which are generally based on variations of the track-while-scan concept. These advantages include a) enhancement of the signal-to-noise ratio for improved target detection performance, b) a capability to track very large numbers of moving objects in the field-of-view, c) improved tracking of merging/crossing trajectories and object clusters, and d) robust performance in stressing scenarios with high background clutter and temporary loss of data. In addition, the velocity filter is inherently a bulk signal processing approach that is particularly well-suited to implementation via parallel processing techniques.

Key issues investigated during the Phase I algorithm studies included the following:

a. Determination of theoretical probability of detection and false-alarm rate as a function of target signal-to-noise ratio, image sequence length, etc.;
b. Extension of the basic velocity filter algorithms to images with non-homogeneous clutter backgrounds (including both stars and extended bright regions);
c. Sensitivity to various scenario-specific effects including object scintillation, the large dynamic range of scene intensity, target acceleration, and image frame misregistration.

2.1.2 Theoretical Studies. An important part of the Phase I algorithm studies was the derivation of the optimum (maximum-likelihood) filter for detecting a target moving through a sequence of N passive sensor image frames with a known, assumed or estimated vector velocity. The target signal and noise observations in the image pixels were modeled as independent Poisson processes to account for the statistical nature of the photodetection process. For the case of a uniform noise background (due to either homogeneous infrared clutter or detector noise), it was shown that the optimum detection algorithm for N consecutive image frames could be implemented as follows:

a) subtract the mean noise level from every pixel of each image frame;
b) shift-and-add the N frames according to the target velocity measured in pixels per frame;
c) correlate the combined frame with the spatial target signal;
d) threshold the result.

For the point-like objects of primary interest in SDI broad-area surveillance applications, the spatial signal in c) is simply a "streak" whose shape and orientation depend on the sensor point-spread function and dwell time and the apparent target velocity in the azimuth and elevation dimensions. For a given sensor, the processing steps in b) and c) above are completely characterized by the target vector velocity in the scene; hence the term "velocity filter."

These basic analytical results have recently been extended to scenes with spatially nonhomogeneous random backgrounds due to the presence of strong IR point sources (such as stars) or extended clutter regions (e.g., clouds) in

the sensor field-of-view. For this more general case, the optimum velocity filter implementation calls for a slight modification of the background suppression step in a) above. Specifically, after removal of the spatially nonstationary mean μ_{ij} from each pixel (i,j), the image cell observations should also be divided by their respective noise variances (or powers) σ_{ij}^2. This latter operation is best viewed as two successive normalizations of each scene by its spatially-varying noise standard deviation σ_{ij}. The first division by the σ_{ij} converts the variable scene background to an essentially homogeneous image against which the moving targets are more readily detected. The latter normalization by σ_{ij} weights the target responses (if any are present) in proportion to their average amplitude signal-to-noise ratio in each image pixel. This choice of weights maximizes the output signal-to-noise ratio (SNR) of any linear combination of noisy target observations from multiple frames, each of which (in general) has different average SNR due to the variable background.

The overall detection performance of the velocity filter is dependent on an output signal-to-noise power ratio (SNR) which was shown to be proportional to the square of the average target signal power (expressed as a mean electron rate at the detectors), the integration period or dwell time on target, and the total number of frames N; and inversely proportional to the apparent target velocity in image cells per dwell period. A perfectly matched velocity filter always provides an SNR gain of N relative to the average SNR available on a single image frame. Mismatches between the actual target velocity and the assumed filter velocity cause losses in the filter output SNR. Such losses, which can be calculated for both the scanning and staring sensor cases, are an important consideration in the design of finite velocity filter banks used for acquiring objects over a wide range of possible velocities.

Detection and receiver operating characteristic (ROC) curves for the optimum velocity filter are shown in Figures 2 and 3. These results are based on a Gaussian approximation to the underlying Poisson image statistics and are valid for images with relatively large photoelectron counts in each cell. Figure 2 shows the probability of detection vs. average input SNR per frame for various numbers of image frames N at false alarm probability 10^{-6}. The ROC curves in Figures 3a-c show that reliable target detection/acquisition performance can be achieved by filtering relatively small numbers of image frames, even at low input SNR on the order of 0-6 dB per frame.

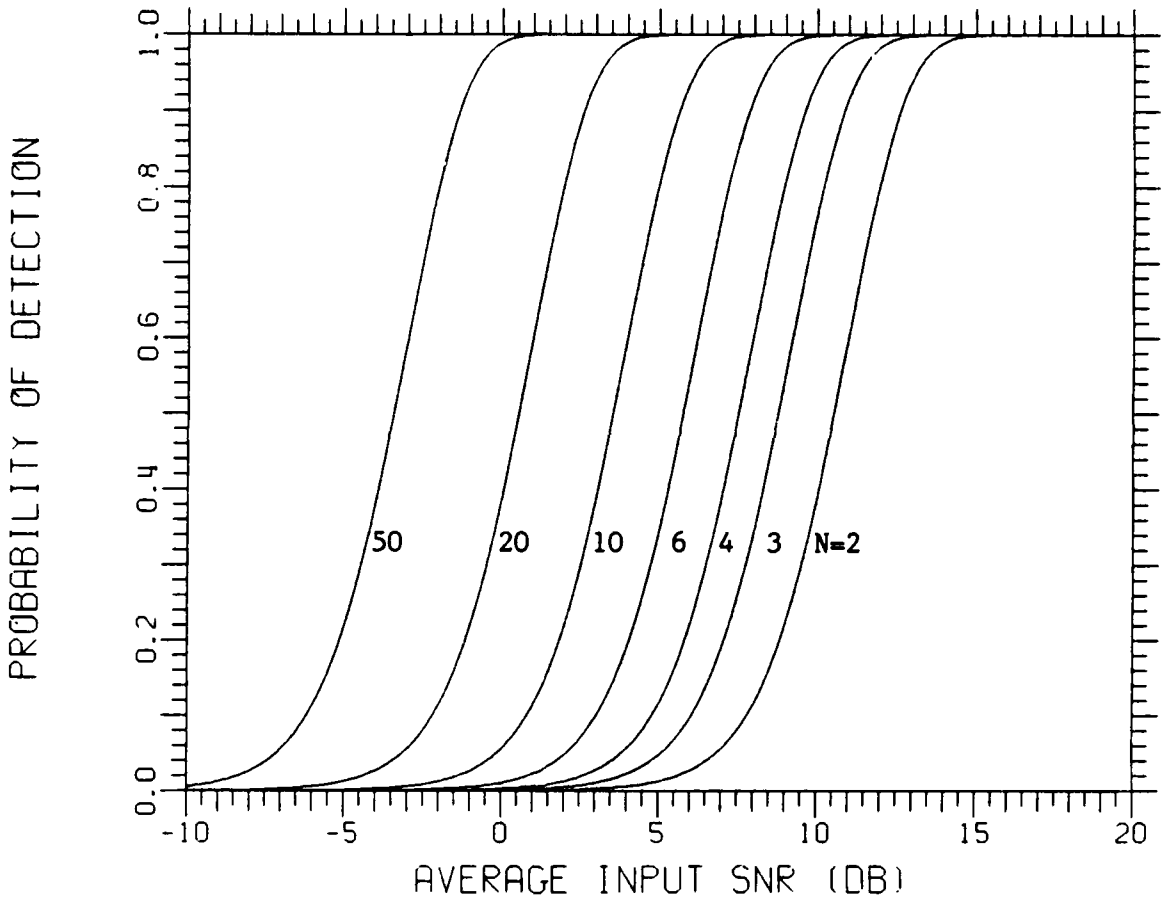

Figure 2. Probability of Detection vs. Input SNR (False Alarm Probability 10^{-6}).

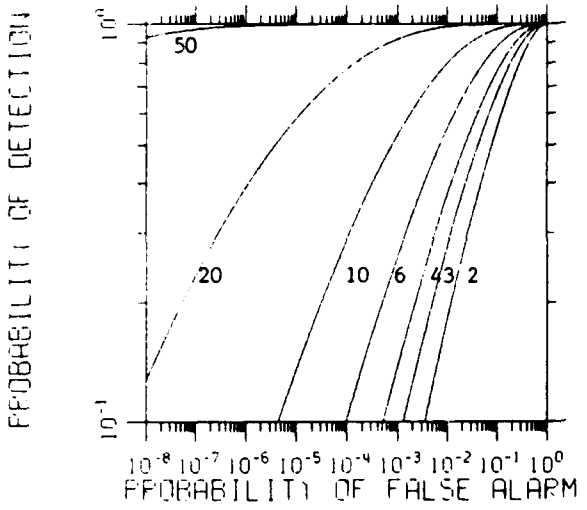

Figure 3a. ROC for 0dB Input SNR.

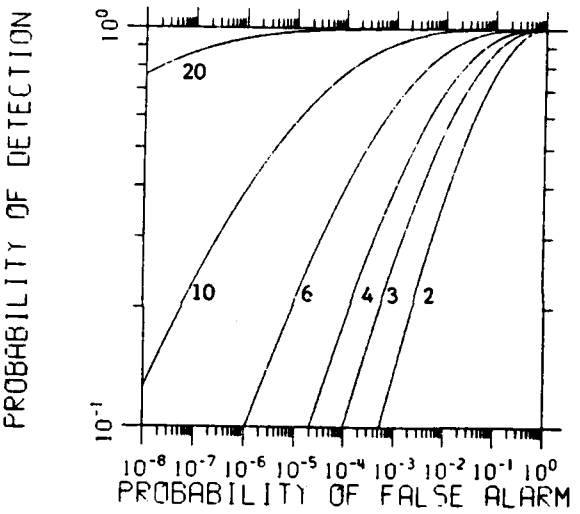

Figure 3b. ROC for 3dB Input SNR.

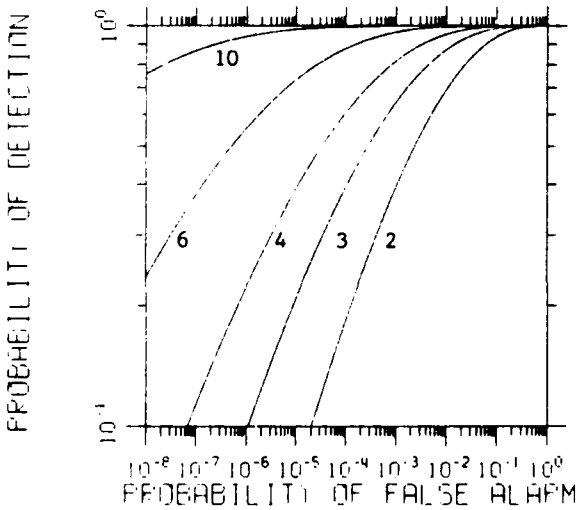

Figure 3c. ROC for 6dB Input SNR.

2.1.3 Experimental Studies. Algorithm simulation experiments were conducted on real and simulated passive sensor imagery to validate some of the basic aspects of velocity filter detection and tracking performance. A variety of target and background image scenes were processed, including a subset of the 1995 Centerline MEA-1 Threat from TRW, a 112-target set used for scan-to-scan correlation studies at the MIT Lincoln Laboratory, and several sets of digital imagery recorded by the Lincoln Laboratory 30-inch "Quad-Camera" optical telescope in New Mexico. The latter source of imagery was of particular interest since it was collected on an actual CCD focal plane array. The Quad-Camera datasets we processed in Phase I included a sequence of scenes of a fast-moving satellite; these were used for testing the optimum velocity filter. Scenes of the Great Nebula in the constellation Orion were also used to study background suppression. All of these scenes exhibited the effects of CCD photodetector noise in addition to a number of other artifacts including image jitter, clipping due to limited dynamic range, and "holes" caused by dead detector cells. Some experiments on the various MIT image datasets are described in the next several sections.

2.1.4 Velocity Filter Detection of a Satellite Streak. The application of velocity filtering to the detection of moving objects in passive sensor imagery was investigated by processing a sequence of six staring telescope scenes plotted in 16-level gray scale format in Figure 4a. An orbiting object appears as a streak in each of the images due to the finite frame integration time. The physical mechanism that caused the brightness fluctuation is not known, but the composite image of Figure 4b clearly shows it to be a periodic phenomenon. Note also the presence of a very bright fixed star and several vertical bands caused by the CCD readout process.

Prior to velocity filtering, we used 5 additional independent images of the same scene background (taken after the target had moved out of the field-of-view) to degrade the average target SNR to about 3 dB in each frame. These low-SNR images are shown in Figure 5. Figures 6a-d show the step-by-step results of velocity filter processing on input frame number 6 in Figure 6a. Figure 6b is the same image after removal of the nonstationary median in each pixel, which was estimated from a total ensemble of 11 sequential images of the same scene. The median filtering operation suppressed the fixed star clutter and other spatially-varying background artifacts, creating a

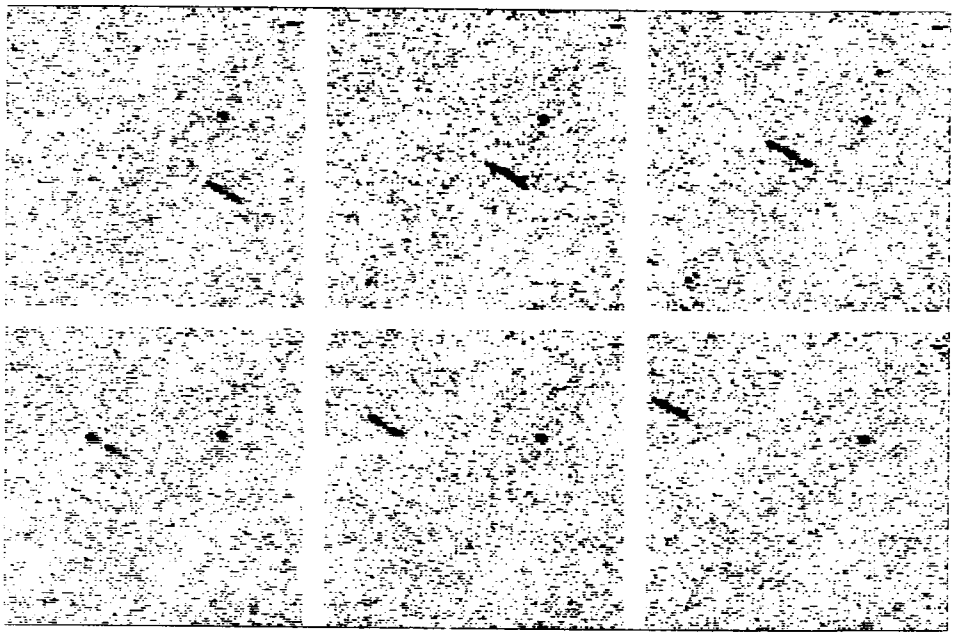

Figure 4a. Six Telescope Scenes with Satellite streak.

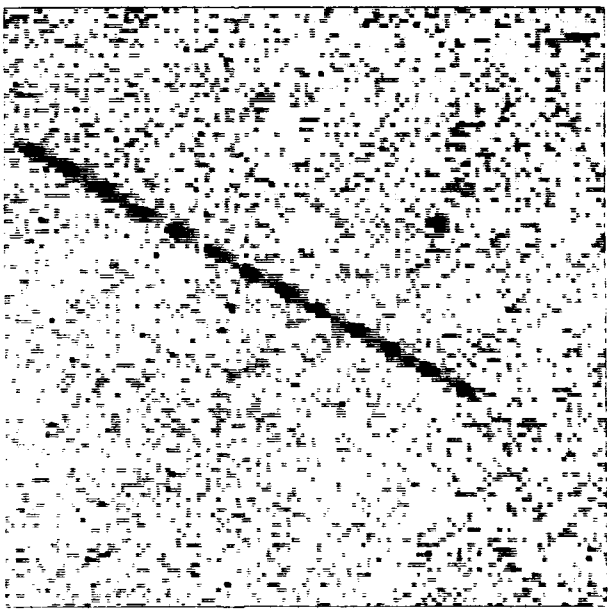

Figure 4b. Composite of Six Scenes in Figure 4a.

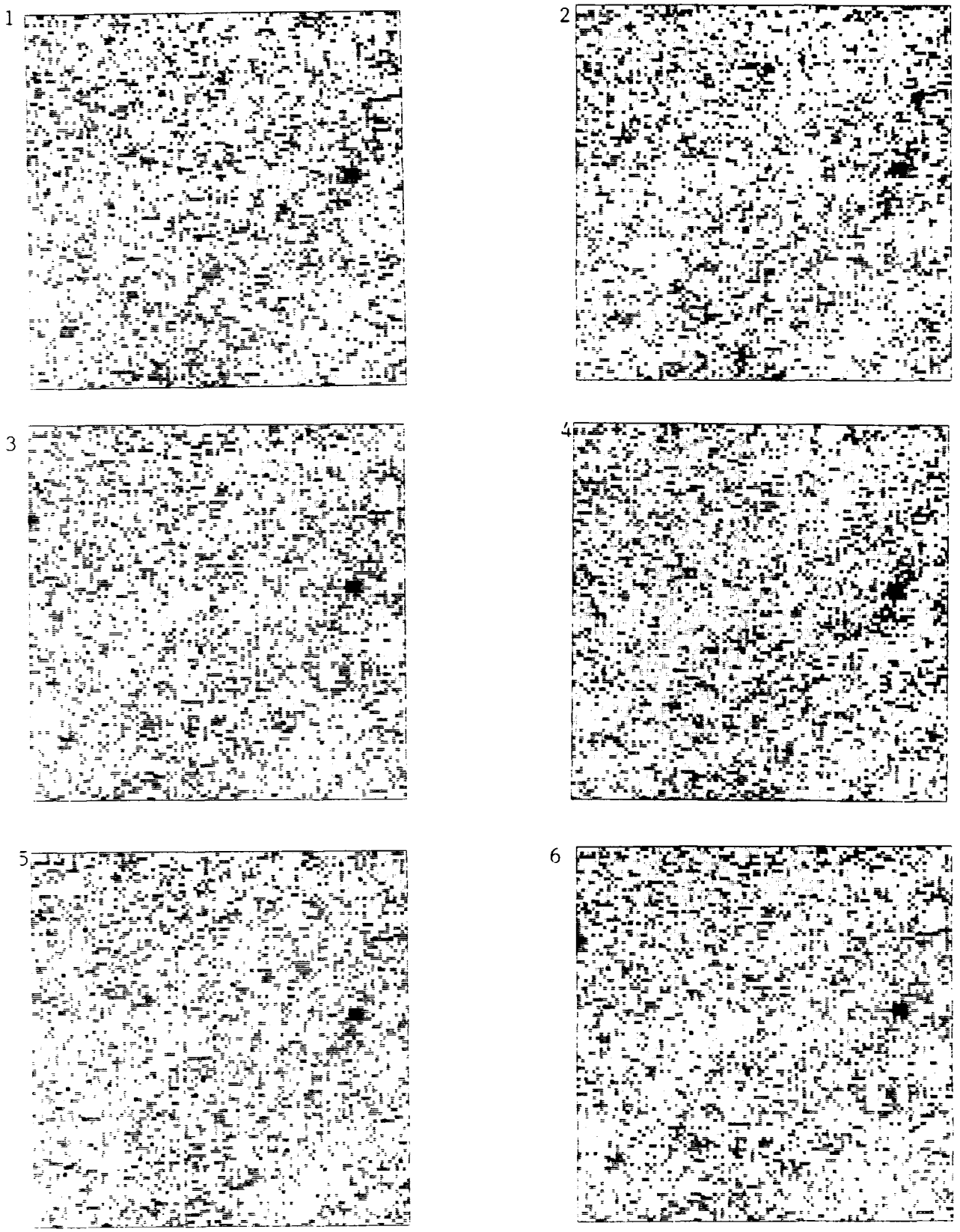

Figure 5. Images with Low Signal-to-Noise Ratio.

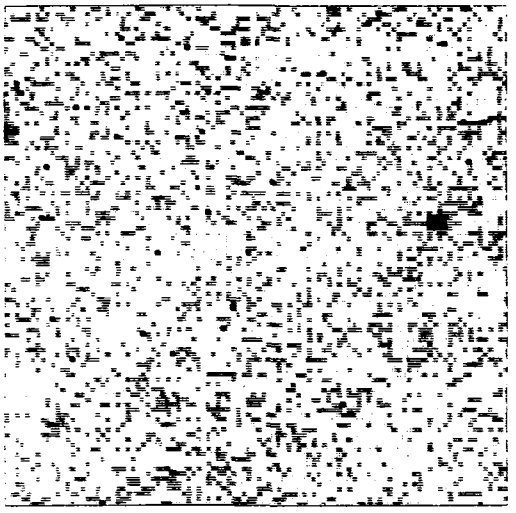

Figure 6a. Input Frame 6.

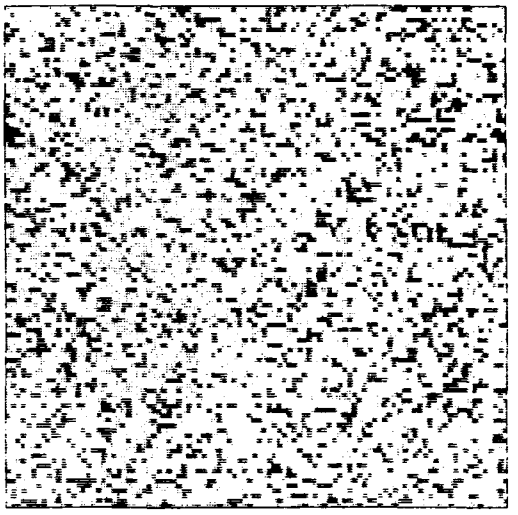

Figure 6b. Frame 6 After Median Removal

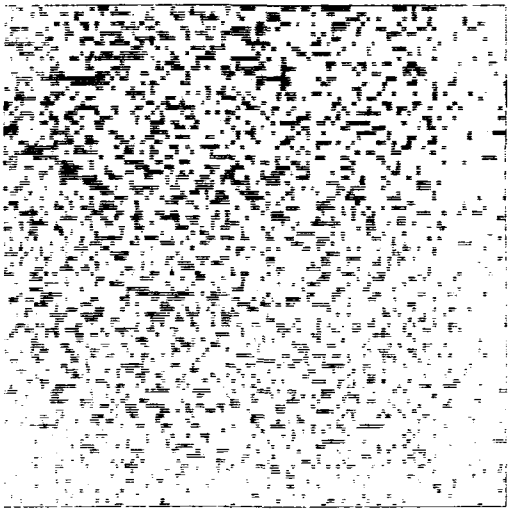

Figure 6c. Frame 6 After Shift-and-Add Velocity Filtering.

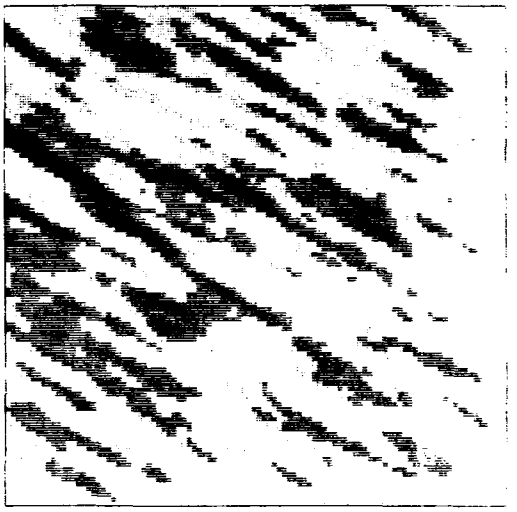

Figure 6d. Frame 6 After Correlation with Streak Function.

homogeneous background. Figure 6c shows the composite frame that resulted from shifting-and-adding the six frames of Figure 5 according to the known target velocity. The result of correlating Figure 6c with a streak template representing the average target signal produced the final output shown in Figure 6d. Comparison of Figure 6d with the original frame 6 image in Figure 4a shows that the moving streak has been successfully integrated out of the background by velocity filter processing. The measured output SNR was about 10 dB, which represents a 7 dB processing gain due to 6-frame velocity filtering.

This example illustrates the potential of the velocity filter to acquire very weak moving objects that could not be detected using ordinary single-frame thresholding or spatial filtering techniques.

2.1.5 Background Clutter Suppression. As explained in Section 2.1.2, the first step in optimum velocity filter processing is the suppression and normalization of non-homogeneous clutter. The Phase I clutter suppression experiments were conducted on the 11-frame sequence of MIT Quad-Camera 120x120 pixel images of the Great Nebula in Orion, which are plotted in Figure 7. The histogram of an 11th scene in the sequence is shown in Figure 8 to indicate the typical range of image pixel magnitudes prior to processing.

For optimum clutter suppression, we would ideally subtract out the nonstationary image mean from each pixel of every frame. Of course, in practice the true mean values are not known a priori and must be estimated from the available data. We tested two different methods of mean estimation during Phase I; one based on sliding-window spatial averages in each frame and another based on time averaging across like-indexed cells of the different frames. The latter approach was found to be superior for the Orion dataset due to the extreme spatial variability of the background.

The results of mean removal applied to the raw data frames of Figure 7 are shown in Figure 9 (the dynamic range plotted here was reduced by a factor of about 40 relative to the plots of Figure 7 in order to show sufficient image detail). The uneven results seen in the images of Figure 9 suggested that the original data frames were not correctly registered. We have empirically found that frame-to-frame jitter larger than a small fraction of a pixel leads to ineffective background suppression. Although in an operational system it might be possible to register successive scenes with measured data

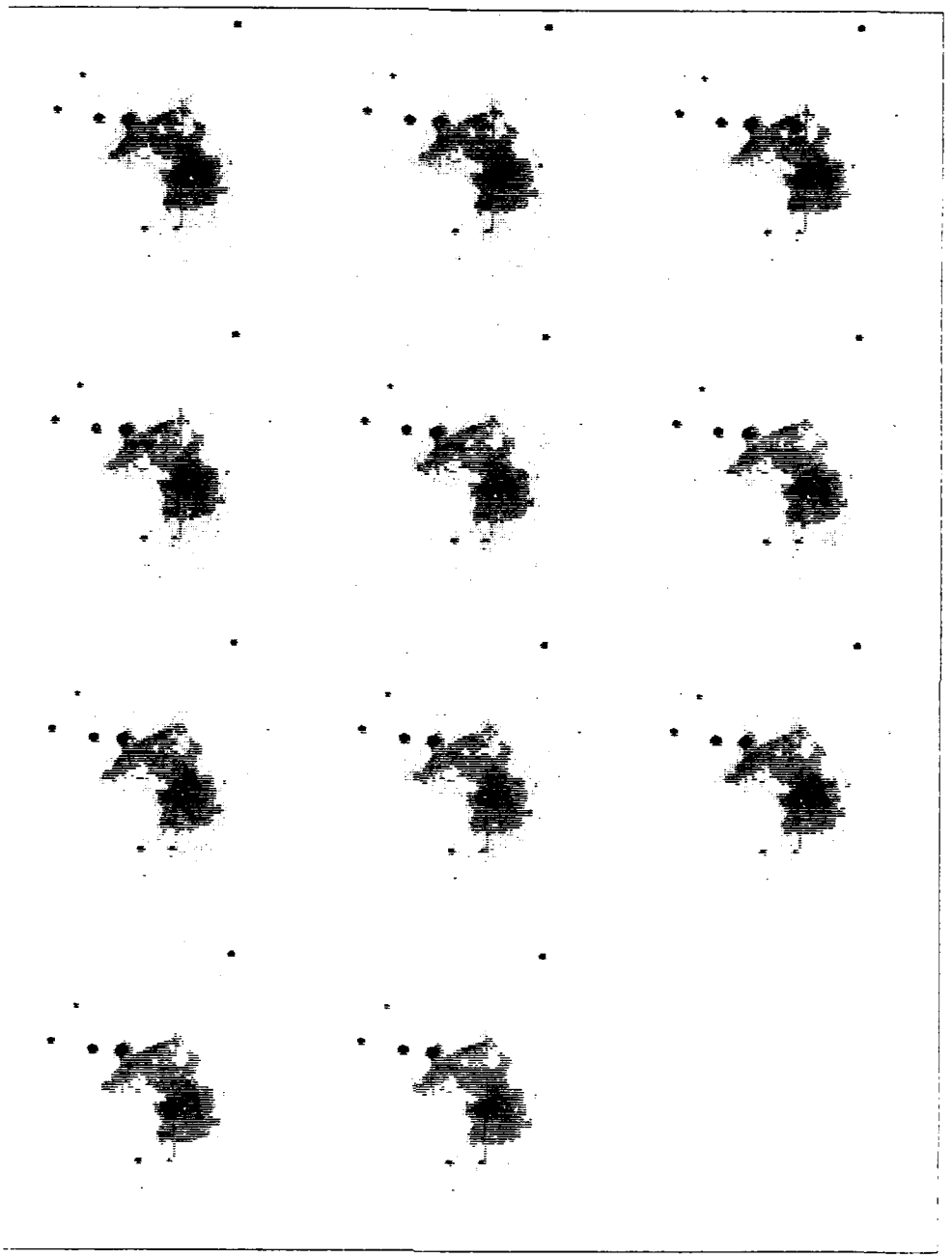

Figure 7. Telescope Images of the Great Nebula in Orion .

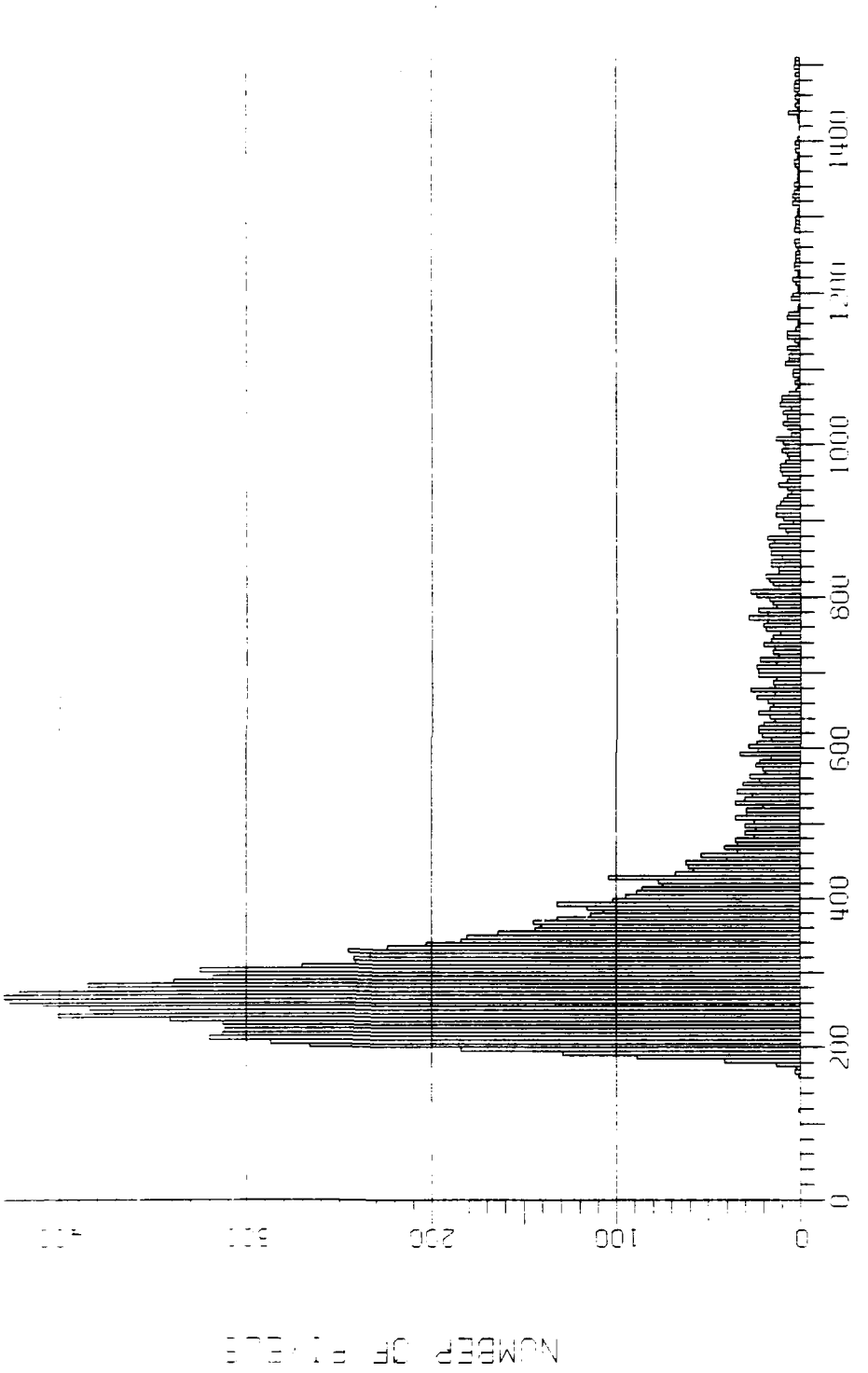

Figure 8. Histogram of Image Frame 11 in Figure 7.

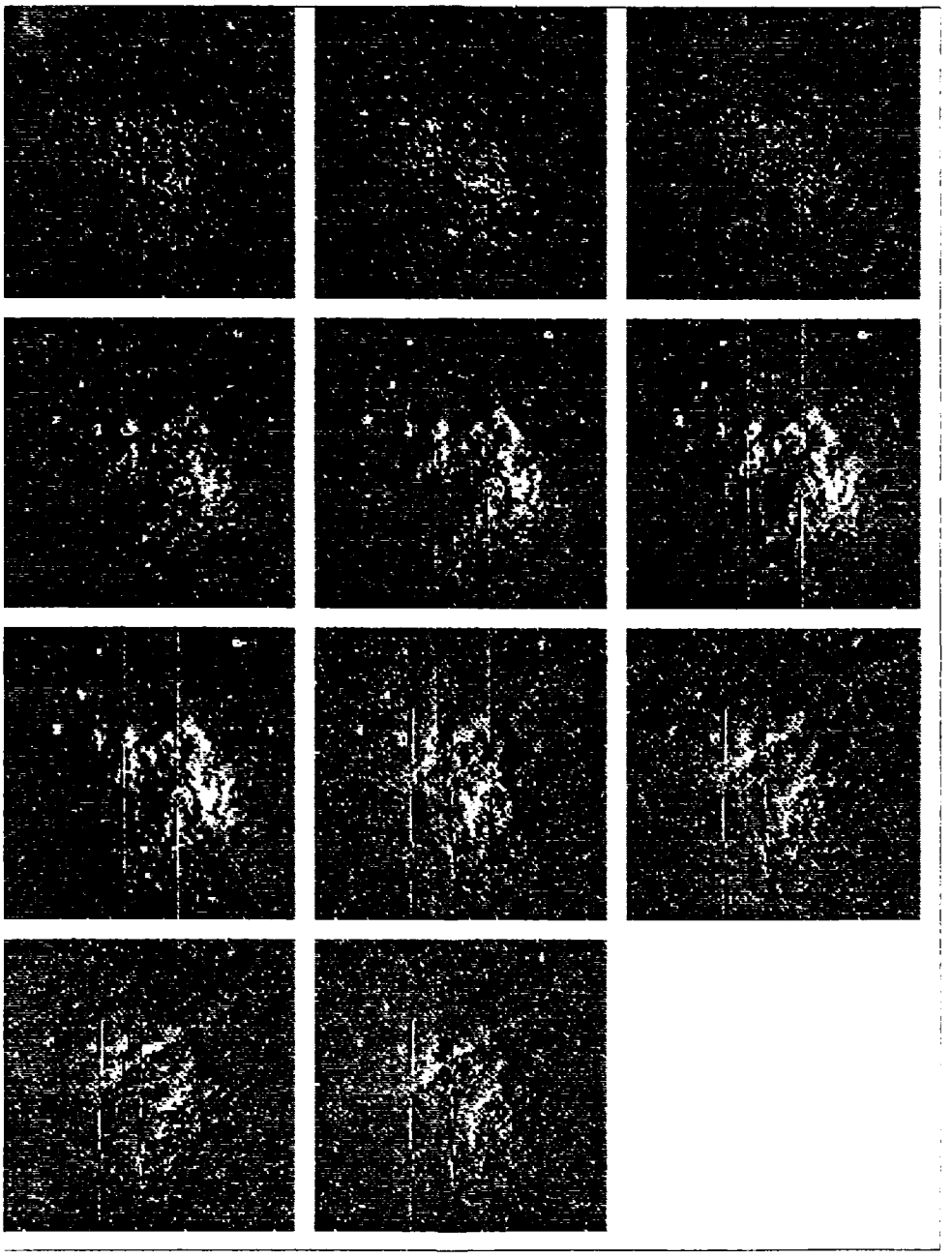

Figure 9. Orion Images with 11-Frame Mean Removed.

from an attitude sensor, in this case we had to self-register the scenes using cross-correlation techniques.

Figure 10 shows misregistration measurements that were obtained by cross-correlating the first frame with each of the others and precisely measuring the zero-offset of the correlation peak in both image dimensions. We then re-registered (or resampled) each original scene by passing it through a spatial interpolation filter having phase shifts proportional to the measured registration errors in each dimension. The results of mean removal on this new sequence of registered scenes are shown in Figure 11 (on the same gray scale used for Figure 9). The registration has eliminated most of the dead cells and readout artifacts (the vertical bands) seen in Figure 9, and the overall suppression is greatly improved. The small loss of image detail in Figure 11 was introduced by the low-pass characteristic of the spatial interpolation filter. Figure 12 shows the histogram of frame number 11 after registration and mean removal. The distribution of image magnitude is now centered on zero and the dynamic range of brightness has been considerably reduced from that shown in the original histogram of Figure 8.

The next stage of background suppression is a normalization by the estimated standard deviation (σ) on a pixel-by-pixel basis. The nonstationary σ was calculated as the square root of the unbiased sample variance in each pixel of the 11 frames. Figure 13 shows the 11 scenes after σ-normalization, and Figure 14 shows the modified histogram of the last frame. These figures show that the clutter suppression processing has transformed the highly non-homogeneous background of the Orion scenes to an essentially uniform noise background with Gaussian statistics. The background suppression operation is analogous to the "whitening prefilter" of linear matched filtering theory, whose role is to create a homogeneous Gaussian noise background against which the conventional matched filter detector is optimal.

2.1.6 Combined Background Suppression and Velocity Filtering. Several experiments were performed on the Orion dataset to evaluate the effectiveness of velocity filtering combined with background suppression in the detection of moving objects. In one such case, we injected a set of 10 relatively weak constant-amplitude point targets in two distinct velocity clusters into the 11 image frames; their initial positions and trajectories are shown superimposed on the background in Figure 15. Figure 16 shows the positions of the target

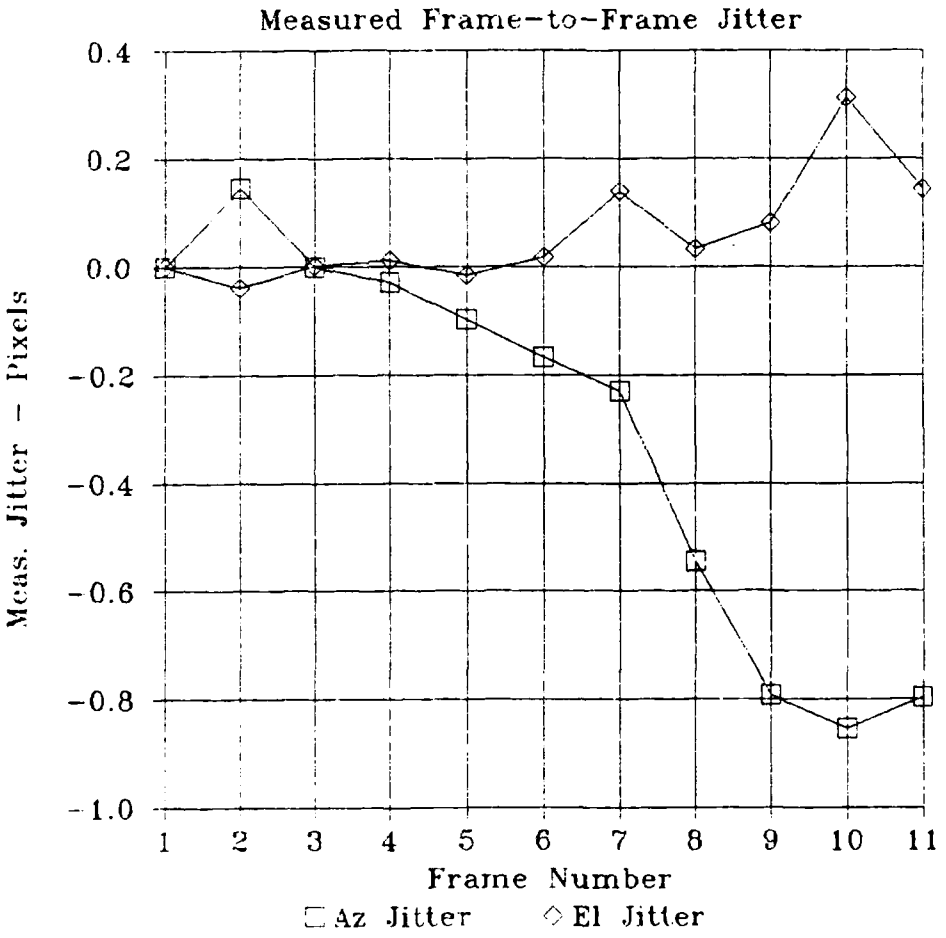

Figure 10. Misregistration Measurements Relative to Frame 1 (Orion Images).

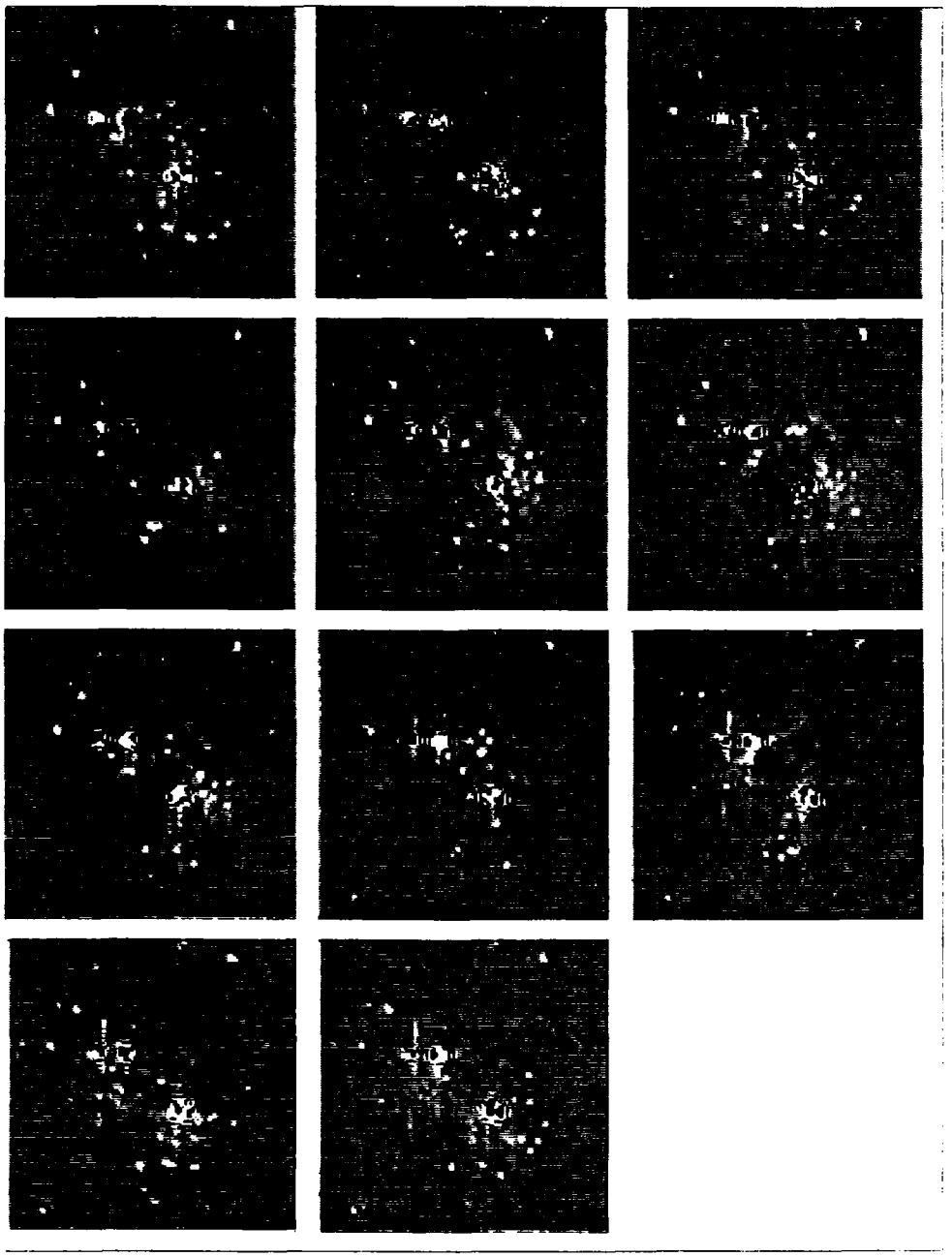

Figure 11. Registered Orion Images with 11-Frame Mean Removed.

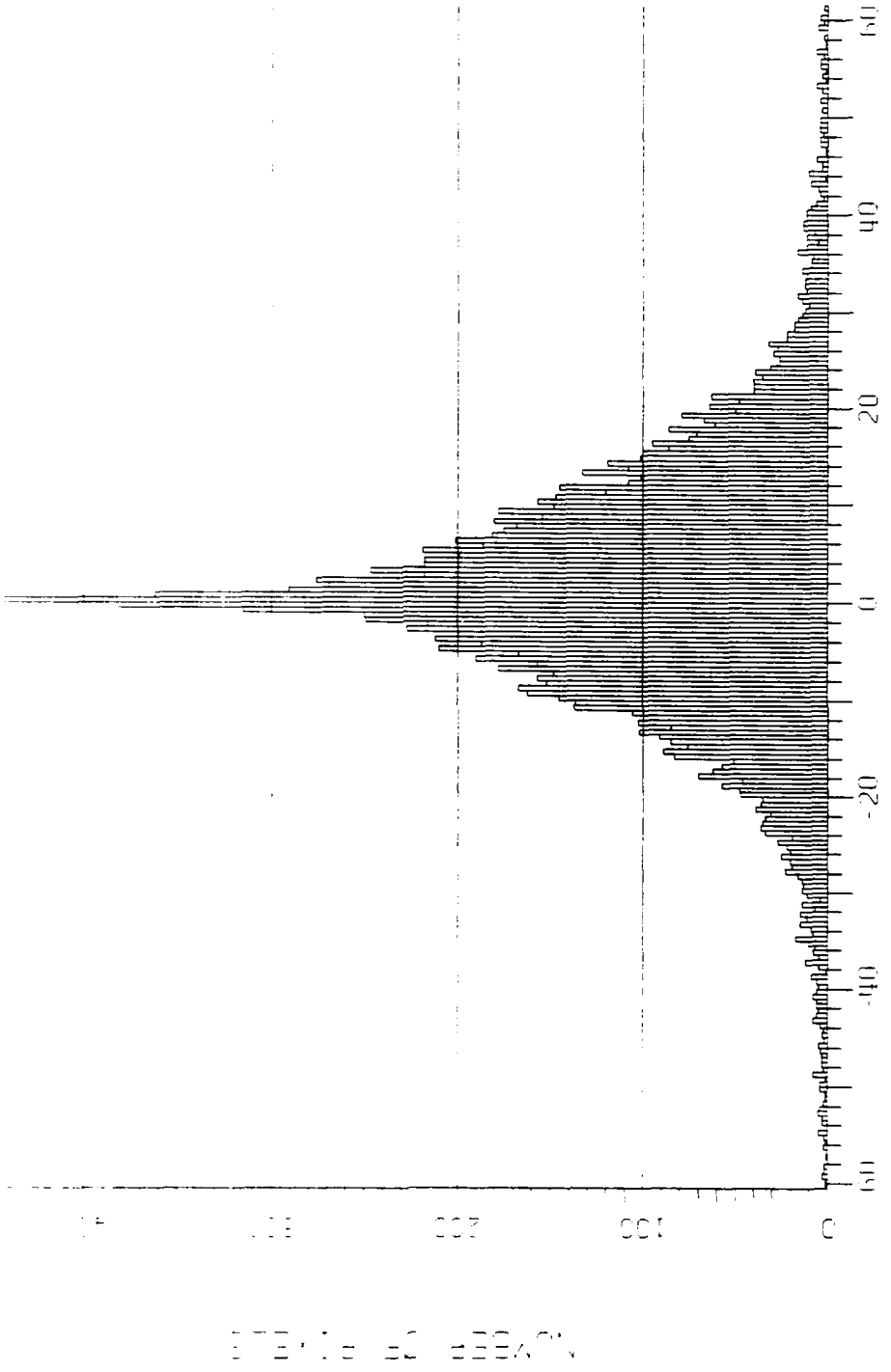

Figure 12. Histogram of Image Frame 11 in Figure 11.

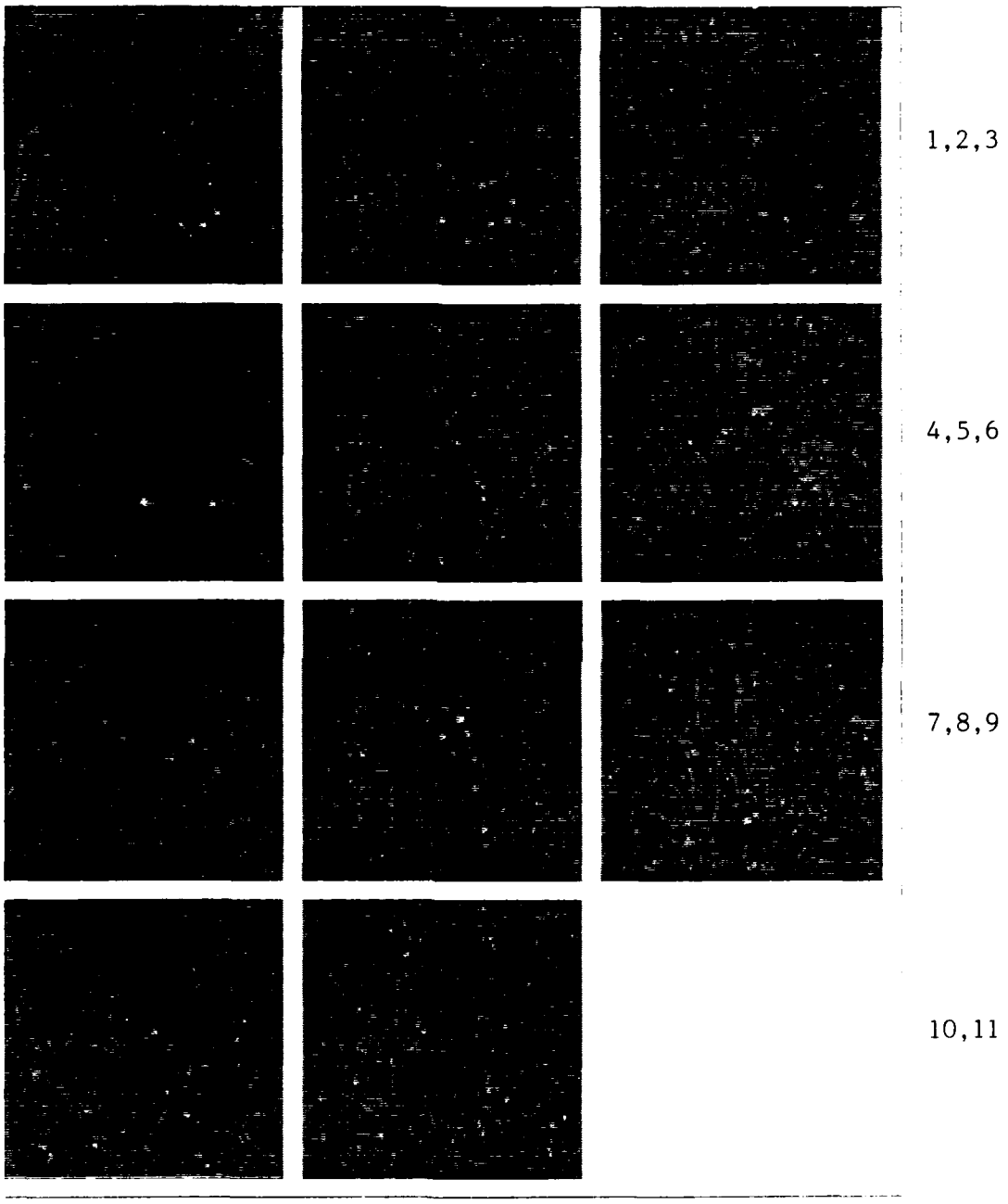

Figure 13. Registered Orion Images After Mean Removal and Normalization by Standard Deviation.

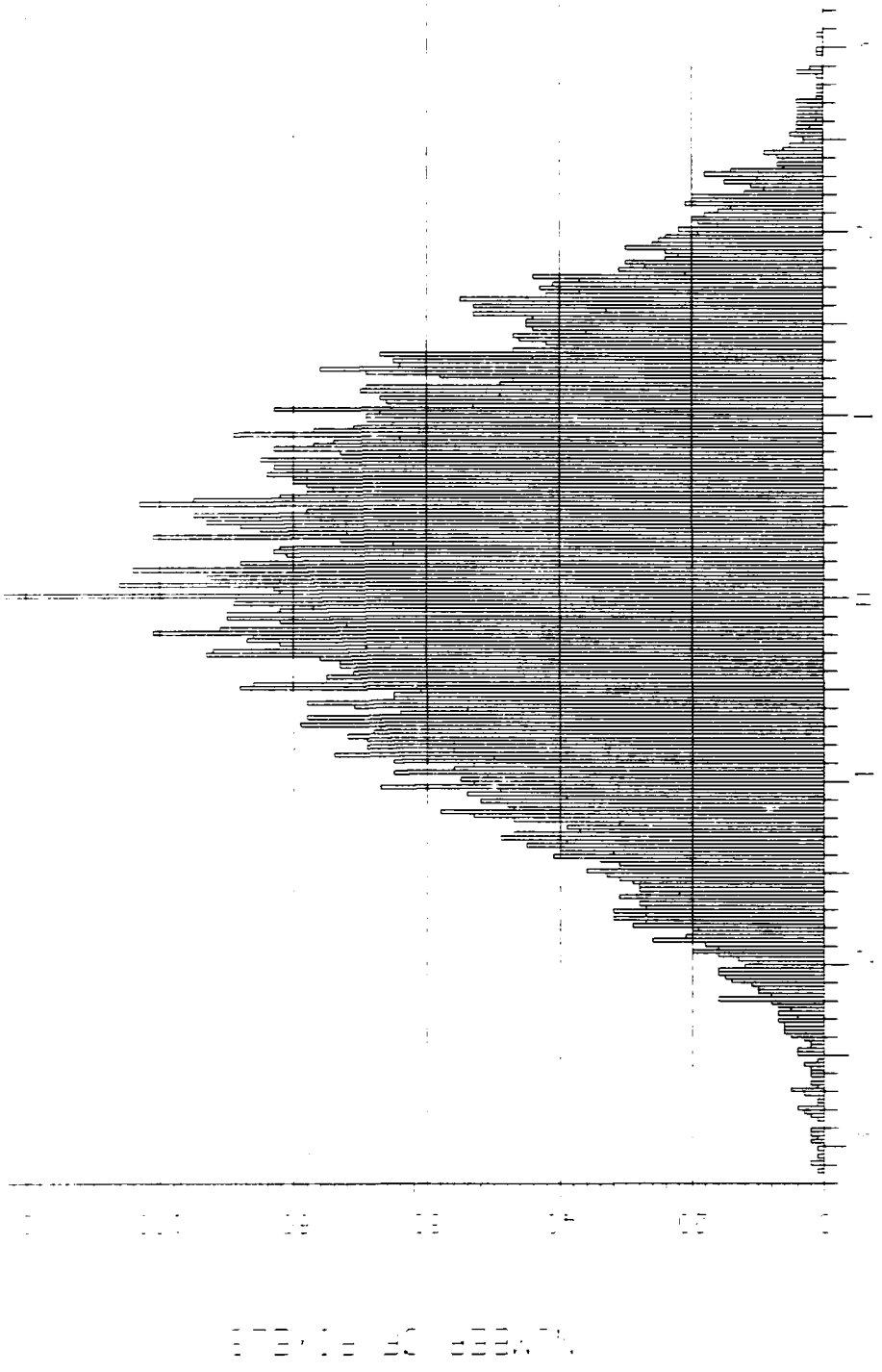

Figure 14. Histogram of Image Frame 11 in Figure 13.

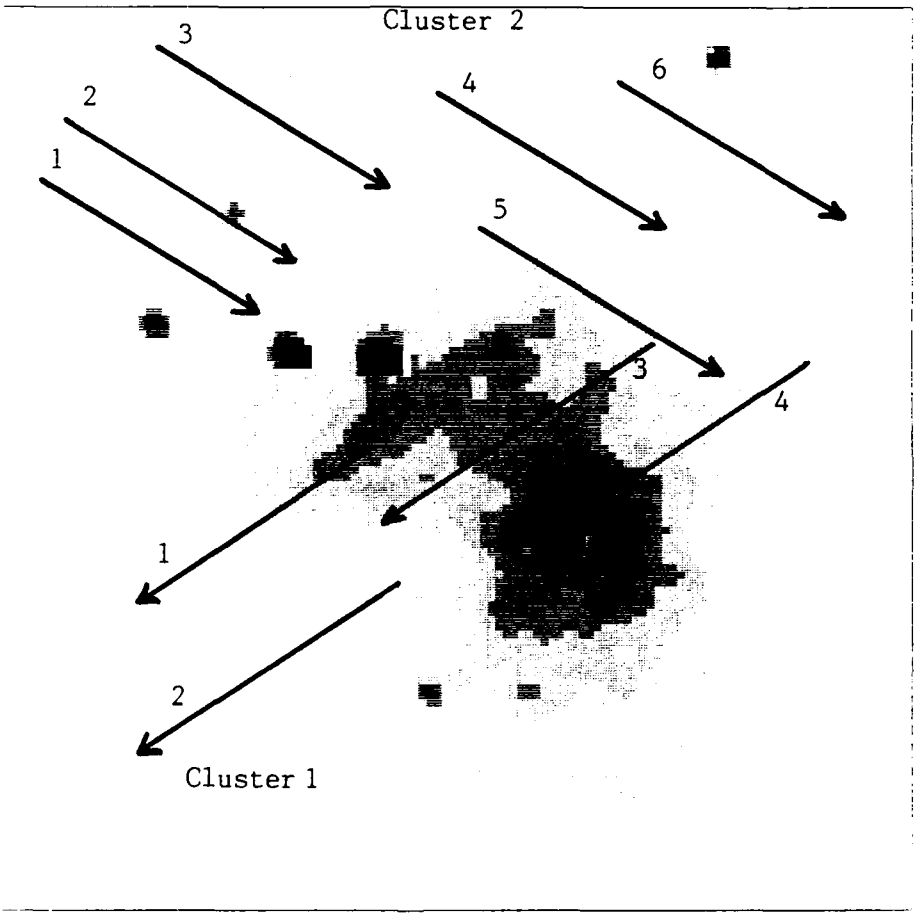

Figure 15. Initial Positions and Trajectories of 10 Targets in Two Velocity Clusters.

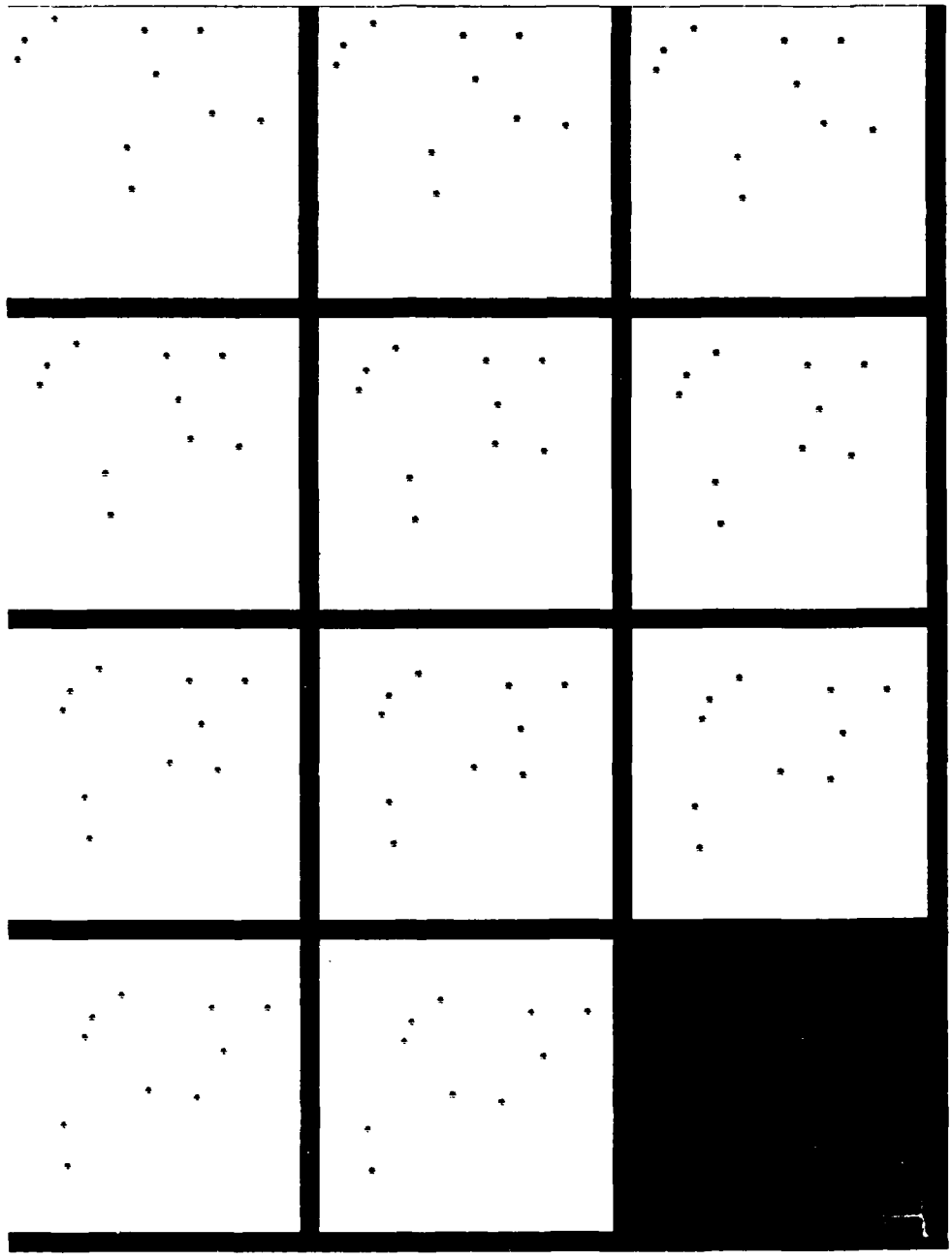

Figure 16. Positions of the Point Target Responses in 11 Frames.

responses in each of the 11 frames. Since each target traversed different portions of the Great Nebula scene, the average input signal-to-noise ratio (SNR) was different for each object as indicated in Table 1 below.

Figure 17 plots the 11 input frames (after image registration) with the synthetic targets included. The same background suppression preprocessing described above was then implemented on each frame. Next, we tested two slightly different versions of the matched velocity filter for each of the target clusters. To implement the optimum detector described in Section 2.1.2, the whitened images were divided a second time by their spatially-varying standard deviations σ prior to the shift-and-add frame integration. As predicted, the optimum filters produced an average SNR gain of 11 (10.41 dB) for each target, with output SNR values as shown in Table 1.

Table 1. Measured Performance of 11-Frame Velocity Filters

Velocity Cluster	Target Number	Average Input SNR	Avg. Output SNR (Optimum Filter)	Avg. Output SNR (Suboptimum Filter)	SNR Loss
1	1	6.04 dB	16.46 dB	15.68 dB	0.78 dB
1	2	6.60	17.01	16.63	0.39
1	3	-0.42	10.00	9.48	0.52
1	4	2.30	12.71	11.53	1.18
2	1	7.70	18.12	17.58	0.54
2	2	5.42	15.83	14.94	0.89
2	3	7.20	17.62	17.30	0.32
2	4	6.55	16.96	16.67	0.29
2	5	4.66	15.08	14.79	0.29
2	6	7.29	17.70	17.35	0.35

The other set of velocity filters were implemented in the same way as the first, except that the second σ-normalization of the 11 scenes was omitted. Table 1 shows that the output SNR for these suboptimum filters was 0.3 to 1.2 dB lower than the optimum, depending on the particular target. However, this particular form of the filter has a very important practical advantage: it requires only a single detection threshold for the entire output frame, rather

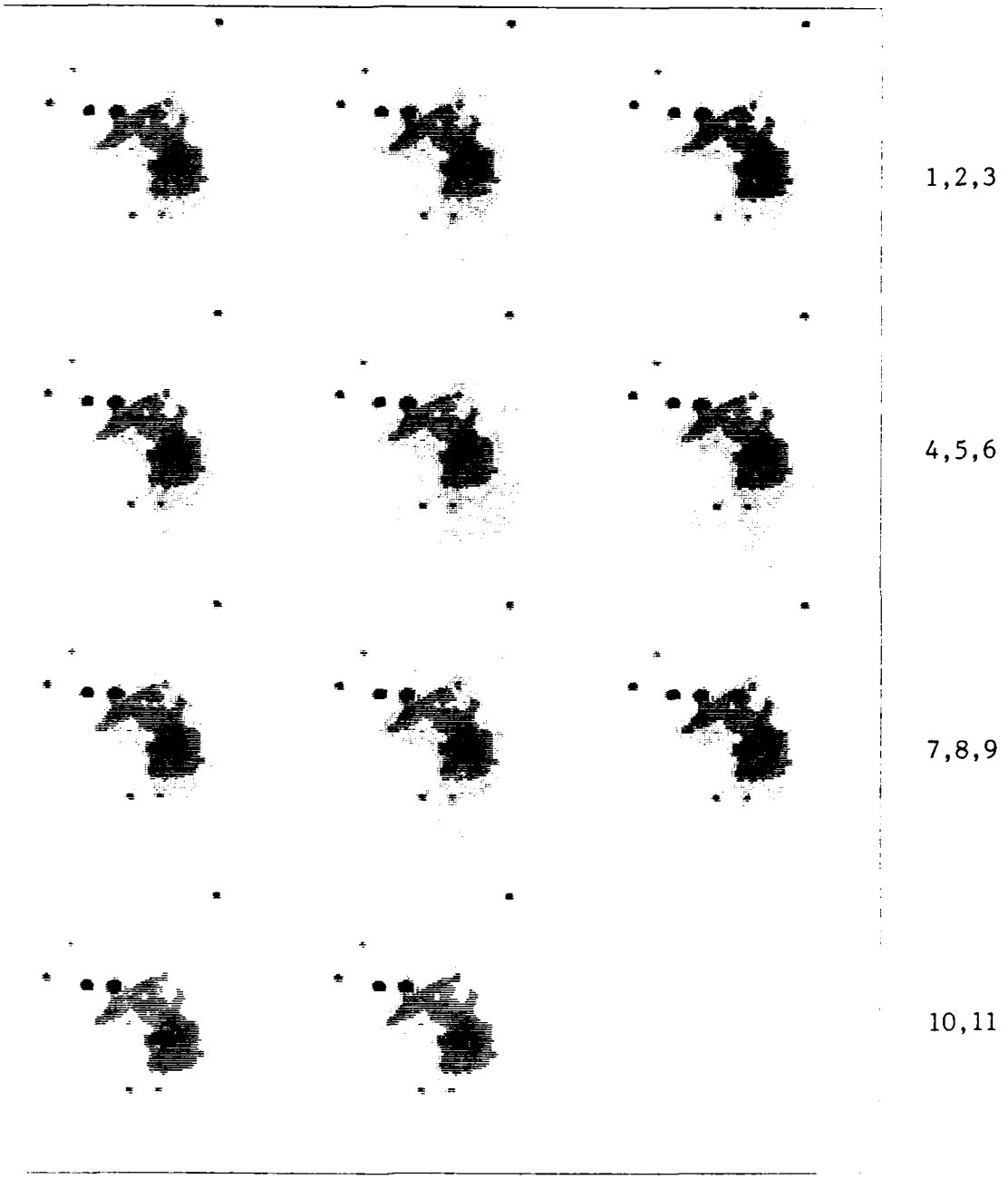

Figure 17. Registered Orion Image Frames with Synthetic Targets Injected.

than a separate one for each pixel. Since the threshold setting in this case does not depend on the input background statistics, the suboptimum form of the velocity filter can provide CFAR (constant false alarm rate) detection performance. The SNR losses shown in the last column of Table 1 represent the cost (in detectability) of CFAR operation for this particular target set in the Great Nebula clutter background.

The output frame for each of the two suboptimum velocity filters is shown in Figures 18a and 18c (referenced to the time of frame 11). Application of a fixed threshold to these frames (at a false alarm probability of around 10^{-6}) produced five detections as shown in Figures 18b and 18d. The five detected targets had measured output SNR levels in the vicinity of 16-17 dB, while the undetected objects (with the unlucky exception of target 1 in cluster 2) had marginal SNRs from 1 to 6 dB lower. To reliably detect the remaining objects, we could integrate more frames in the velocity filters; alternatively we would require a higher input SNR for each object. Figures 19a-d show the results of suboptimum velocity filtering and thresholding for a case where the power of each injected objects was 10 dB higher on each frame. In this case, all 10 point targets were easily detected against the suppressed clutter background.

2.1.7 Acquisition and Tracking of Multiple-Object Clusters. The velocity filter is used to determine the positions of the objects in a sequence of scenes which have a specified two-dimensional velocity. Generalizations of the velocity filter concept can work equally well when the objects have any specified frame-to-frame motion. For objects in a threat cloud the motion must often be described by a polynomial which has terms higher than the velocity term. The acceleration is usually the highest-order term which must be considered, but it is possible to create cases where the next derivative (the jerk) must also be considered. Thus, before the velocity filter can be used it is necessary to determine the significant motion components of the clusters of objects to be acquired for tracking.

Two approaches are available for deciding which velocities to use for the velocity filters. The first is to simply implement a set of filters to accommodate every possible velocity (and acceleration). This "filter bank" approach is practical for reasonably small scenes where the range of possible velocities is not too large, and it constitutes an approximately optimum approach to the detection of moving targets. However, for the applications of

Figure 18a. Velocity Filter Output for Target Cluster 1.

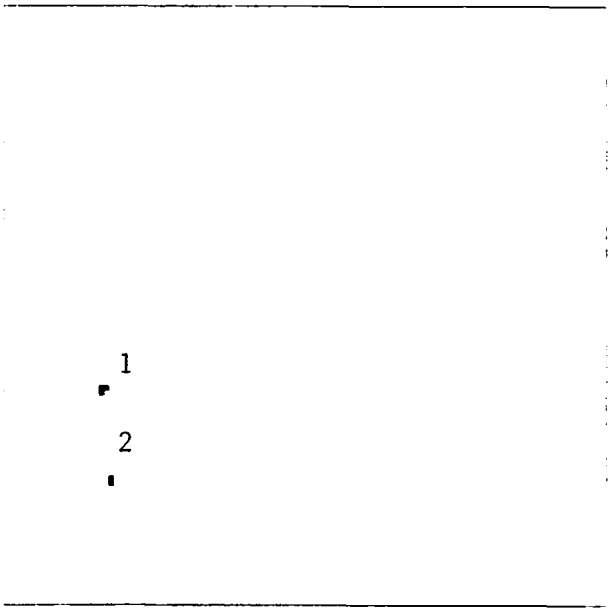

Figure 18b. Detections Obtained from the Velocity Filter Output of Figure 18a.

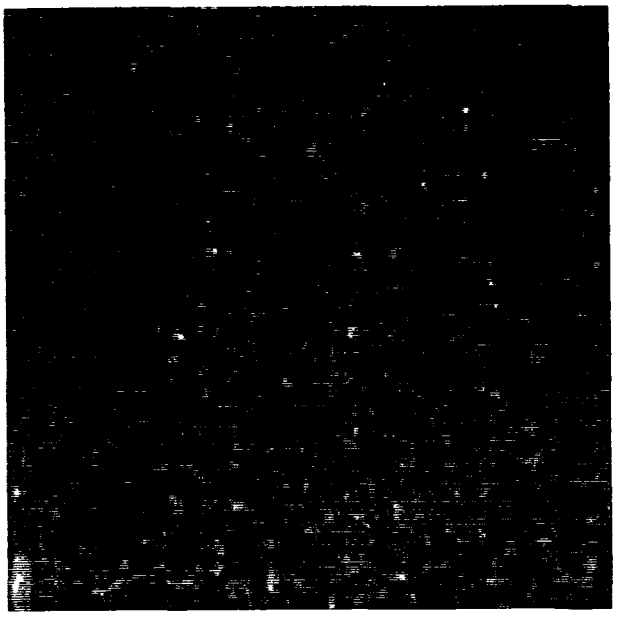

Figure 18c. Velocity Filter Output for Target Cluster 2.

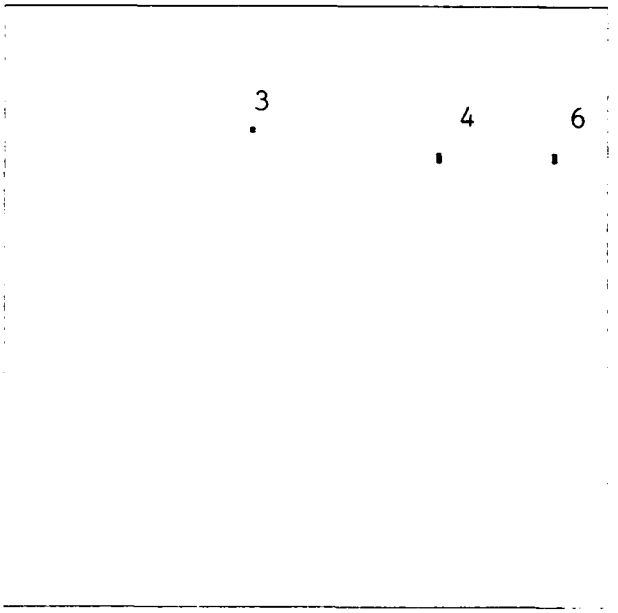

Figure 18d. Detections Obtained from the Velocity Filter Output of Figure 18c.

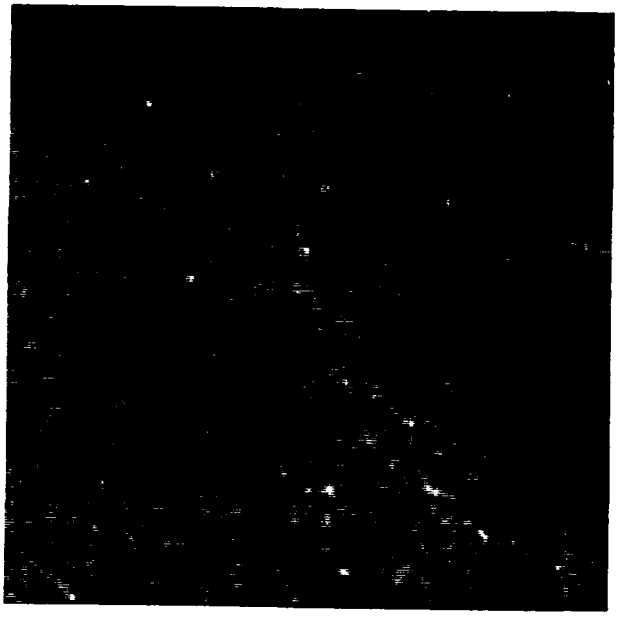

Figure 19a. Velocity Filter Output for Target Cluster 1 (Input SNR 10dB higher than figure 18a).

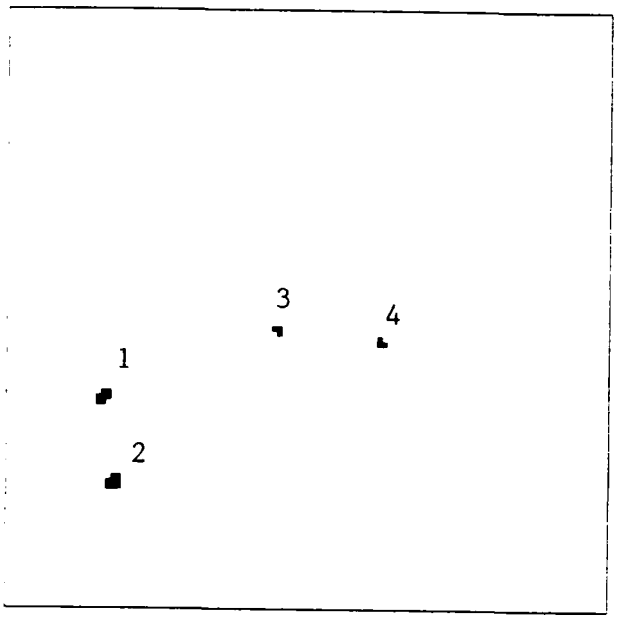

Figure 19b. Detections Obtained From the Velocity Filter Output of Figure 19a.

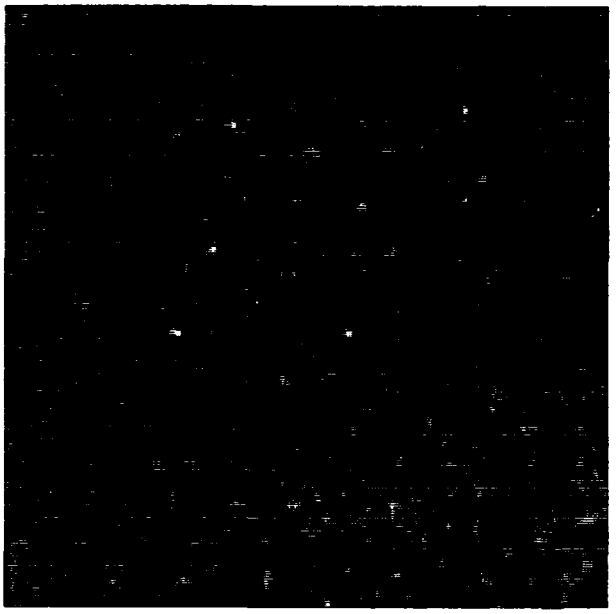

Figure 19c. Velocity Filter Output for Target Cluster 2 (Input SNR 10dB higher than figure 18c).

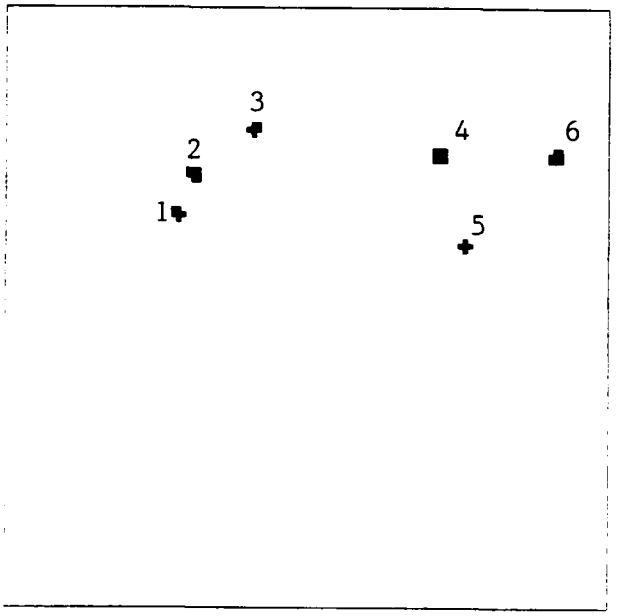

Figure 19d. Detections Obtained from Velocity Filter Output of Figure 19c.

interest here the number of velocity-acceleration pairs which must be considered is much too large to use this approach.

The other approach is to estimate the the velocities and accelerations of the objects in the scene without first determining the positions of those objects. While this sounds like a daunting task, it can be done in a rather straightforward way using the cross-correlation functions of the pairs of available scenes. Here we will illustrate how this is done for scenes created from simulated tracks provided to us by MIT Lincoln Laboratory.

The simulated data represent the output from a scanning sensor, and consist of the locations (azimuth and elevation) of detected responses. Most of these are due to clusters of CSOs. We have converted these data into scene format by placing a Gaussian response with a two-sigma width of 100 μrad at the location of each detection. The result for the first four of 20 frames of data is shown in Figure 20.

The cross-correlation function between two frames measures the similarity of the scenes as a function of the relative shift between them. For N frames of data there are N(N-1)/2 possible pairings, so for the four frames of Figure 20 there are six possible cross-correlation functions to compute. These are shown in Figure 21. In that figure the center of the figure corresponds to no relative shift, and each edge represents a relative shift of the full frame width in that direction. The locations of the various peaks in those functions indicate the relative shifts between clusters of objects in the different frames, and the locations of the highest peaks indicate the relative shifts between the scene pairs as a whole. Knowing the scale factor for the frames in Figure 21 (the full width of each frame corresponds to 22.86 milliradians) and the time between successive frames (10 seconds per frame) it is possible to rescale the scenes of Figure 21 to units of apparent velocity (μrad/sec). The result of doing this is shown in Figure 22, where each cross-correlation function is shown in contour-plot format with the same velocity scale.

Acceleration Estimation. The location of the highest peak in each of the frames of Figure 22 gives an estimate of the velocity of the threat objects averaged over the time separating the two frames. When the locations of these peaks are plotted at the time halfway between the corresponding frame times, the result is as shown in Figure 23. Figure 23a shows the azimuth rate, and Figure 23b shows the elevation rate as a function of time. The slopes of

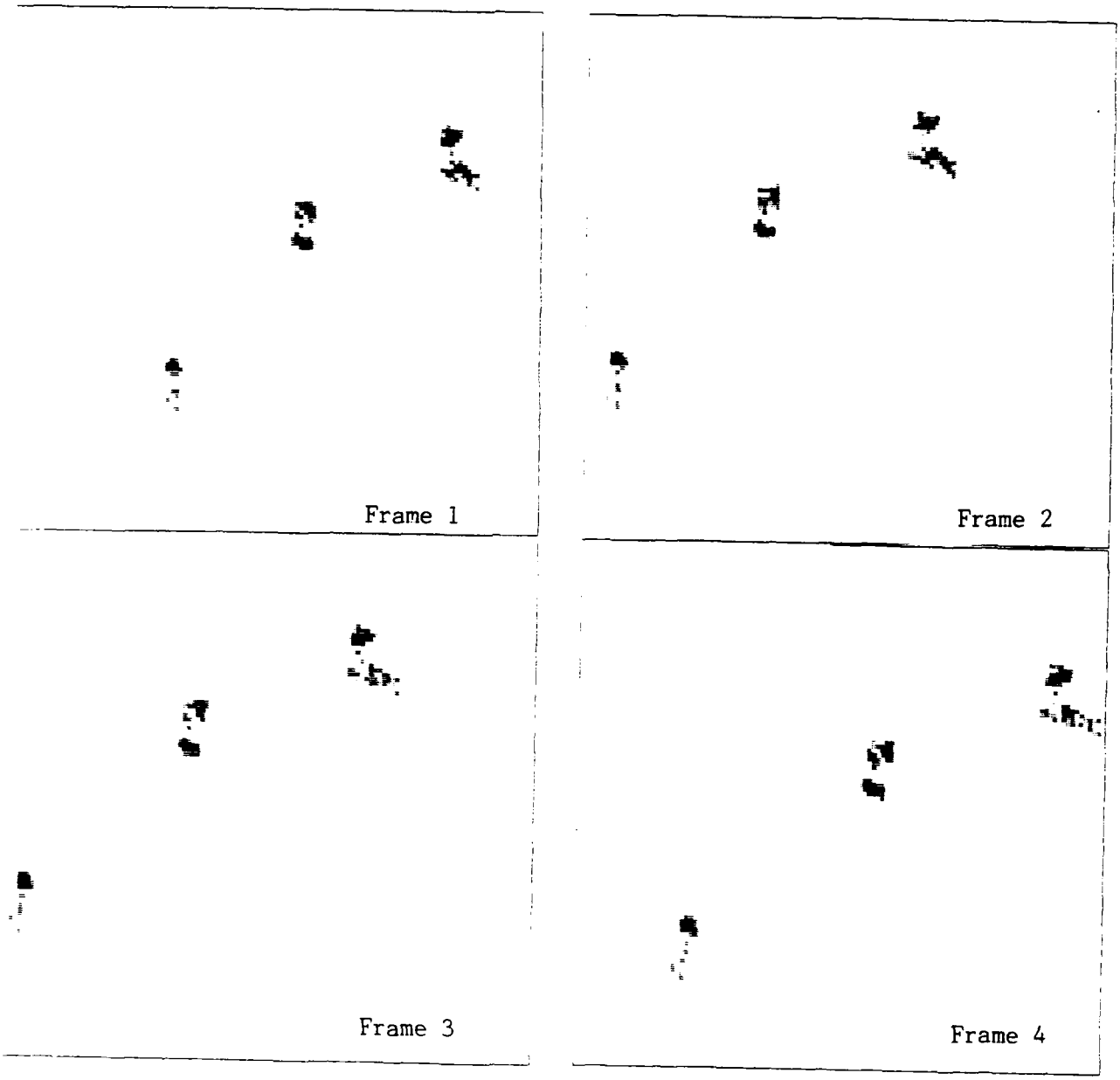

Figure 20. Four Input Scenes

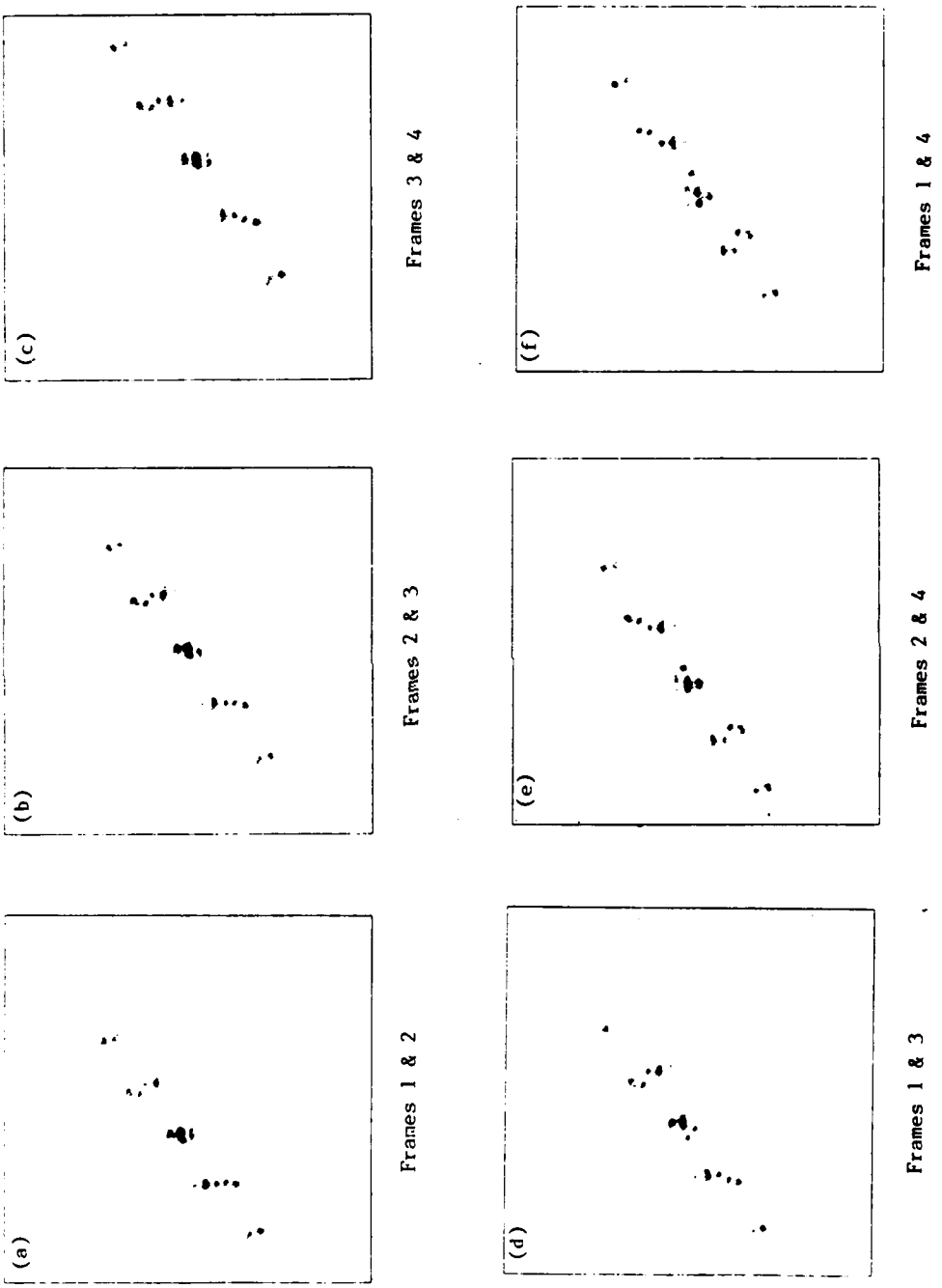

Figure 21. Cross Correlation Between Frame Pairs. (a) Frames 1&2, (b) Frames 2&3, (c) Frames 3&4, (d) Frames 1&3, (e) Frames 2&4, (f) Frames 1&4.

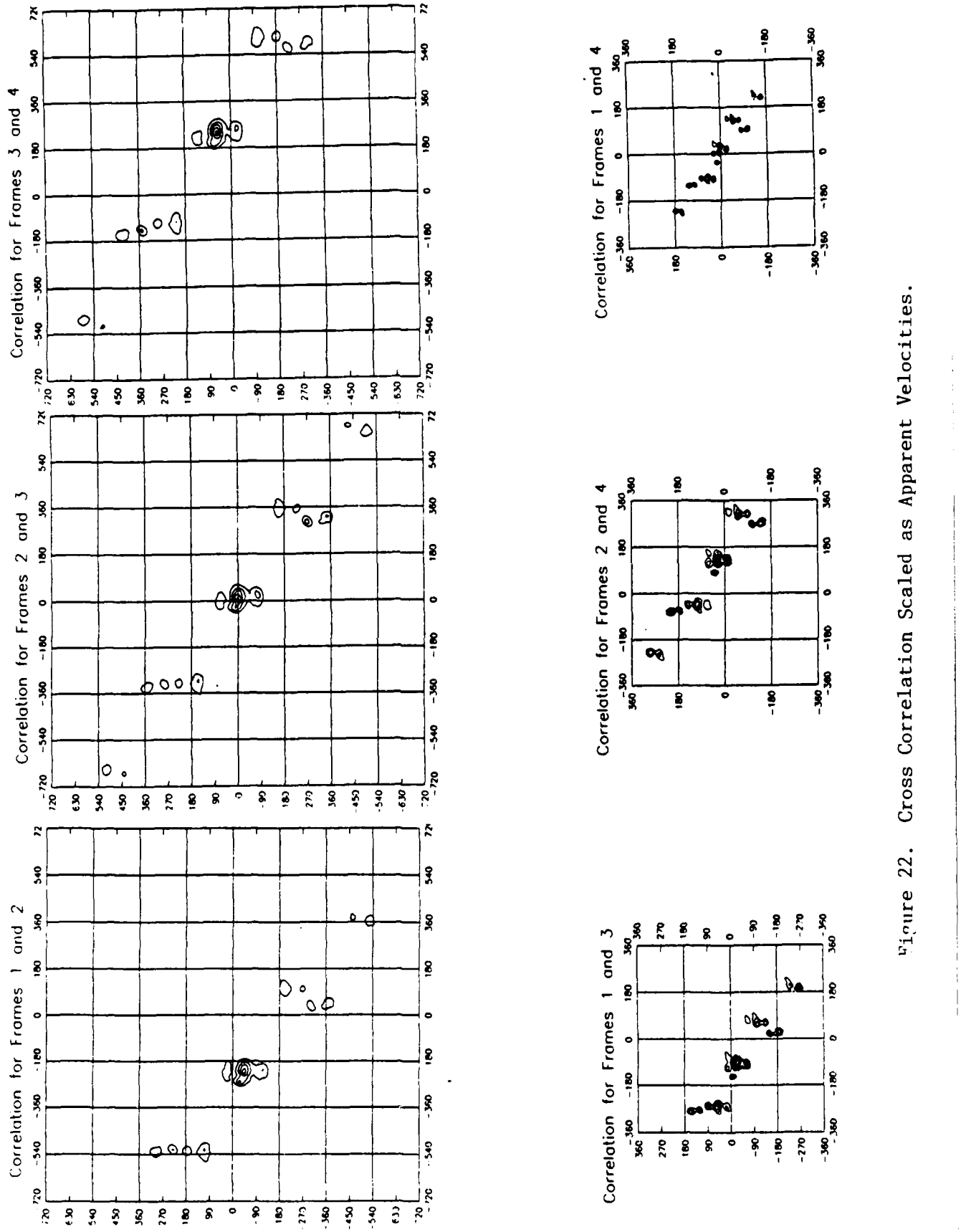

Figure 22. Cross Correlation Scaled as Apparent Velocities.

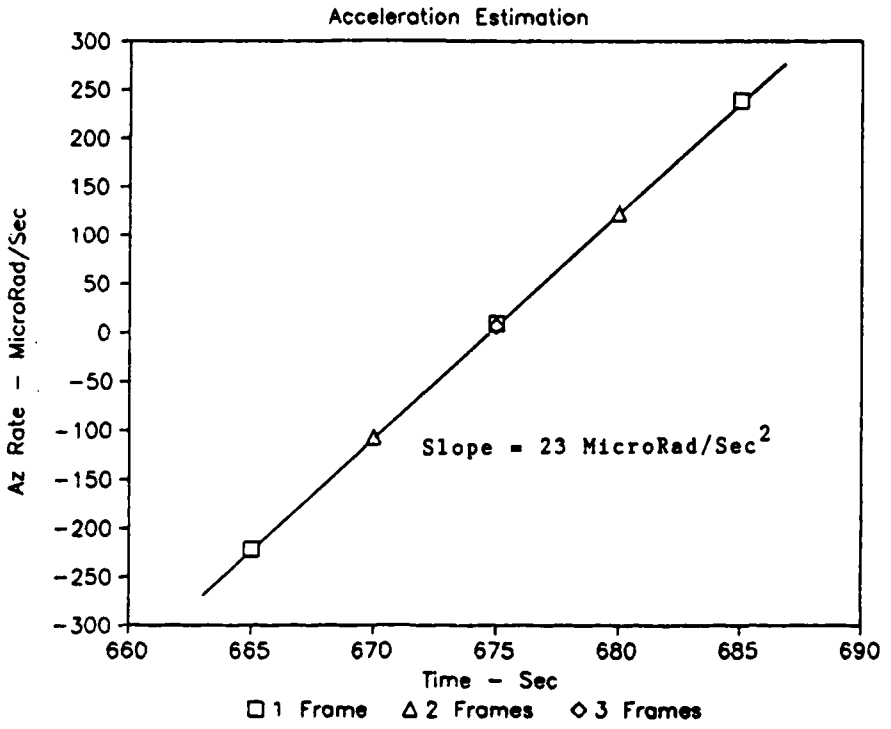

Figure 23a. Azimuth Acceleration Estimate from Correlation-Peak Locations.

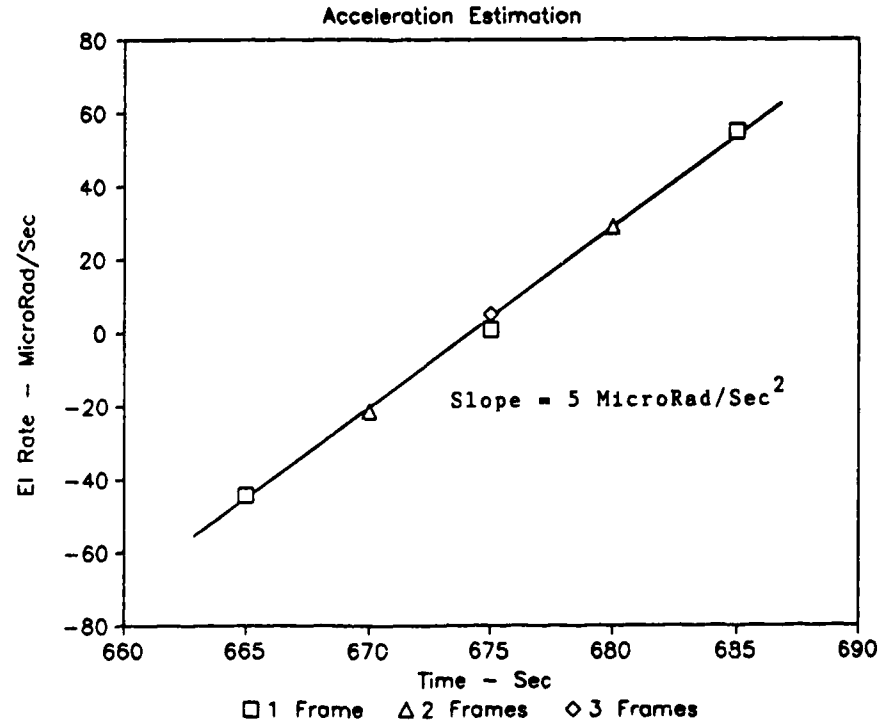

Figure 23b. Elevation Acceleration Estimate from Correlation-Peak Locations.

these curves give the two-dimensional acceleration of the threat as a whole. The fact that the data points fall on a linear curve shows that higher derivatives, such as jerk, need not be considered for these targets.

Velocity Estimation. For the velocity estimate it is necessary to distinguish between the different velocities of the clusters of threat objects. The velocity estimates are obtained by combining the cross correlations of Figure 22 in two steps. The first step is to shift the cross-correlation functions by the amount indicated by the measured two-dimensional acceleration. The second step is to combine the six cross-correlation functions by taking their geometric means, i.e., by multiplying them together pixel-by-pixel, and then taking the sixth root. After the shifting, only the largest central peaks of the cross-correlation functions will coincide. These peaks (before shifting) are shown in Figure 24. The effect of taking the geometric mean is to eliminate all but the central peaks of the correlation between frames one and four (the correlation function with the best resolution and, before taking the geometric mean, with the most ambiguities). The result is shown in Figure 25. The peak to the left at (-32.5, -58.9) μrad/sec is well defined, and clearly indicates the velocity of one cluster of objects in the scenes. The large peak to the right has two subsidiary peaks at (6.8, -70.2) μrad/sec and at (25.1, -75.5) μrad/sec. These indicate the velocities of two other clusters whose velocities are much less well resolved. Under some circumstances it might be necessary to treat them as one cluster moving with one velocity, depending on the resolution of the velocity filters.

Position Estimation. The positions of the objects in the scenes are estimated by passing the four scenes through velocity filters. The four input scenes were shown in Figure 20. Ideally the output of the velocity filter is a copy of the last input frame (frame four in this case) with only the responses for the targets with the selected velocity present. In reality there is some distortion due to relative motion within a cluster of targets, and some responses due to other targets with nearly the same velocity. In other words, the velocity filter has the characteristic of conventional signal filters that the passband is not perfectly flat, resulting in some change to the signal, and the stopband is not perfectly zero, resulting in some residual response from undesired signal components. In Figure 26 we show frame four of the input data and the outputs from the three filters with the velocities identified from the peaks in Figure 25. As can be seen, the output of the

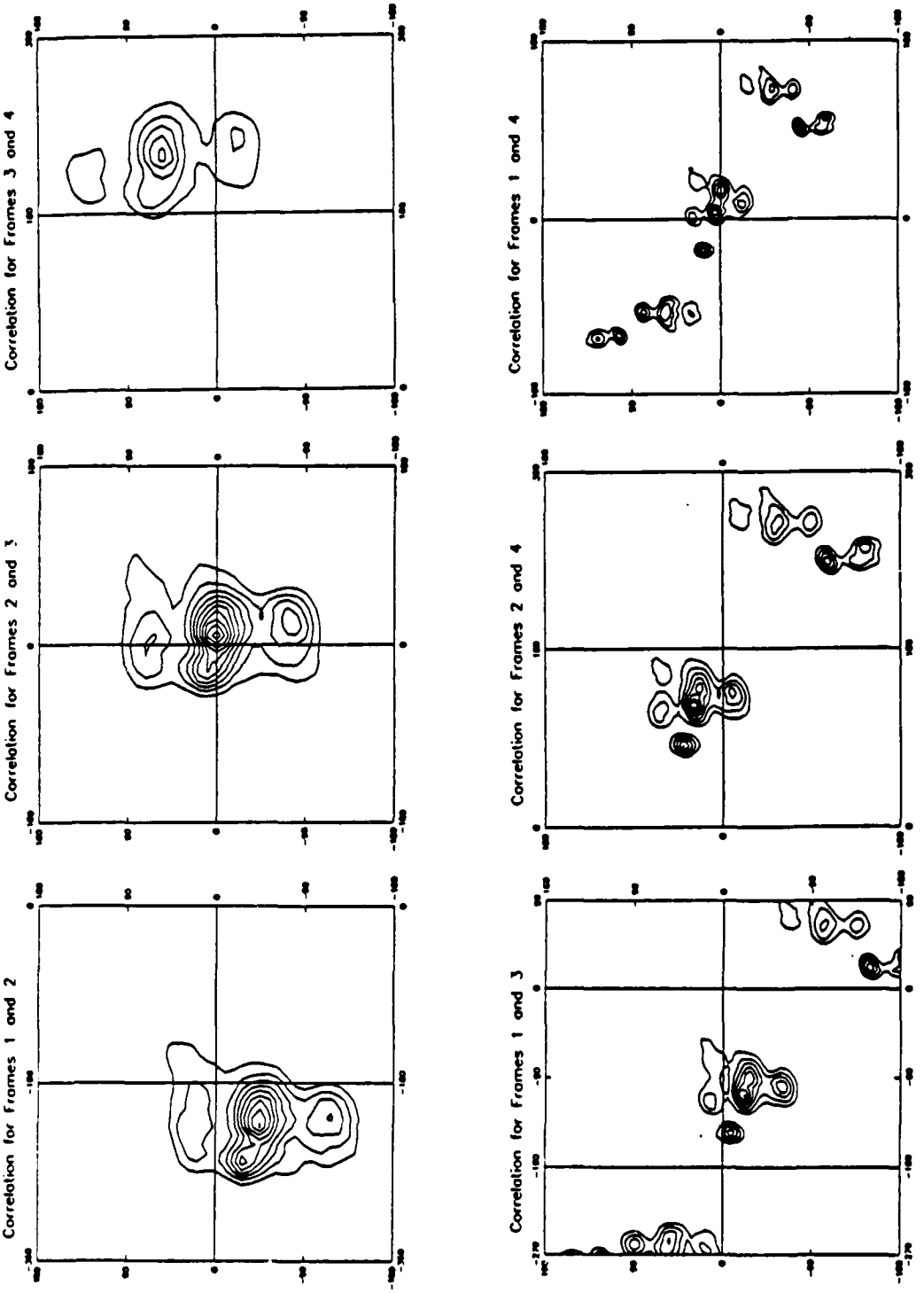

Figure 24. Correlations in Vicinity of Highest Peaks.

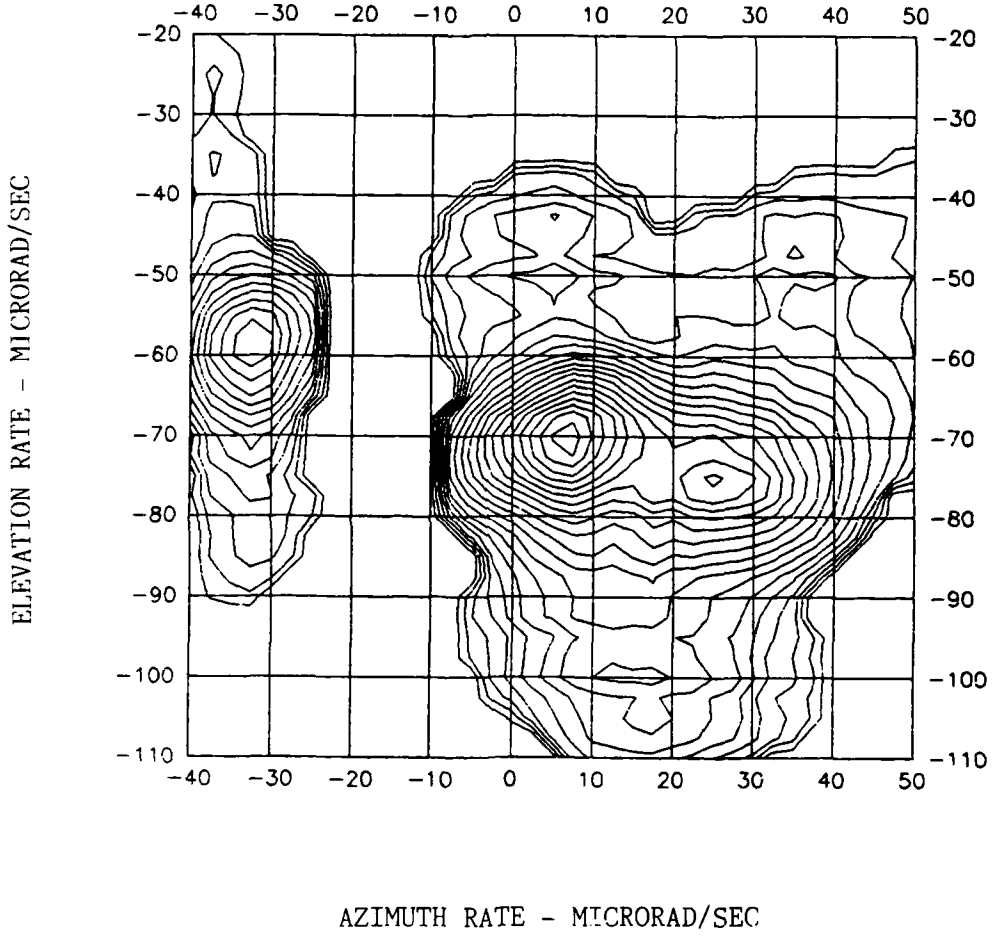

Figure 25. Geometric Mean of Shifted Correlation Functions.

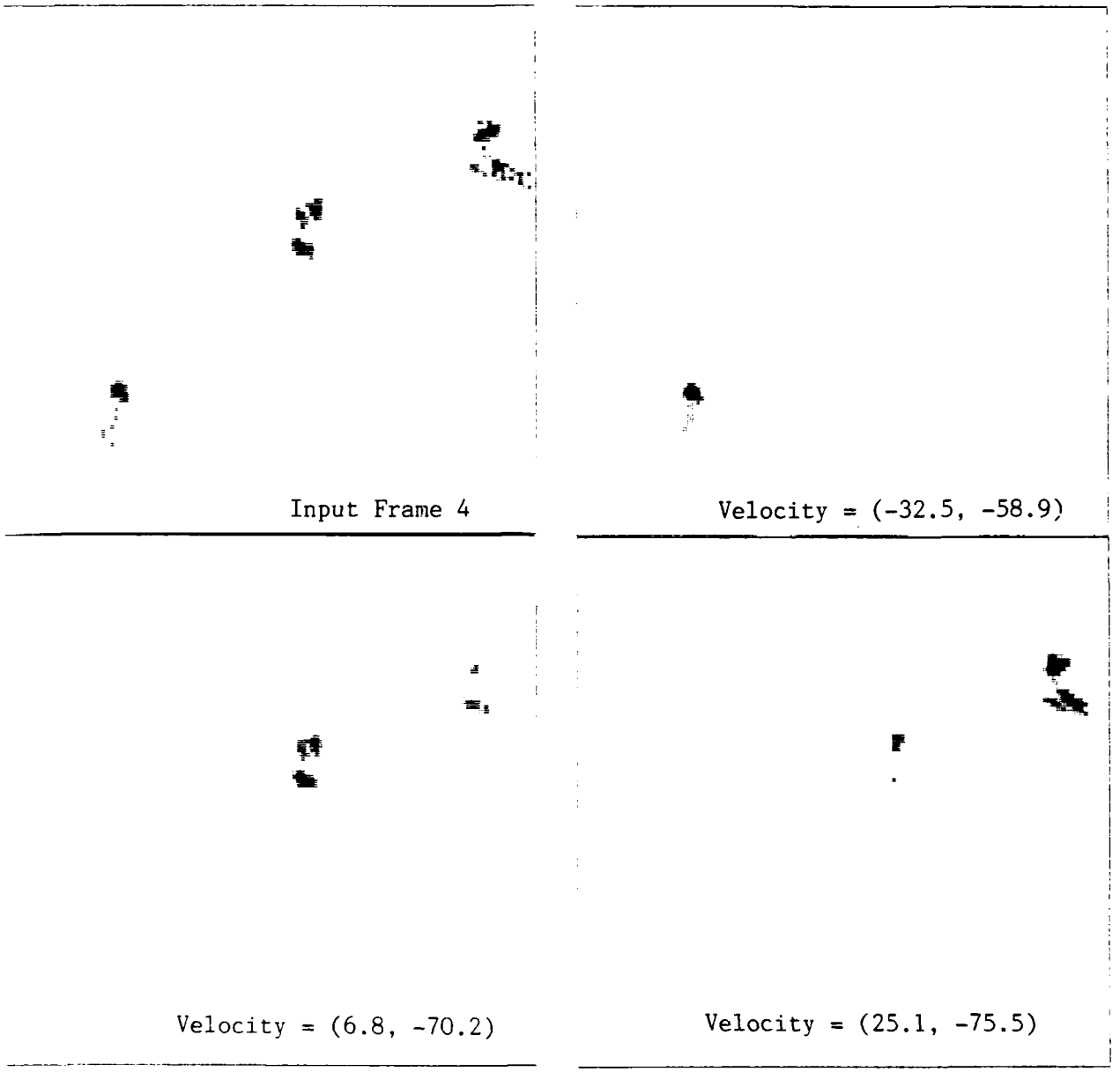

Figure 26. Input Frame Four and Three Velocity Filter Outputs.

filter matched to a velocity of (-32.5, -58.9) μrad/sec is a good replica of the input frame for that target cluster. The outputs from the two other filters matched to (6.8, -70.2) μrad/sec and (25.1, -75.5) μrad/sec and are also good replicas of the input frame for those target clusters, but they have some residual response from each other's targets. This is because their velocities are so close together that they pass some of the responses of each other's targets. As a practical matter this would result in two tracks being initiated for some of the target responses, one track with a velocity corresponding to (6.8, -70.2) μrad/sec, and another with a velocity corresponding to (25.1, -75.5) μrad/sec. Of course, on subsequent frames one of these tracks would be discarded.

2.2 Object Discrimination.

2.2.1 First Stage Discrimination. As noted in Section 3, evidence is accumulating that booster fragments and kinetic kill debris may have very rapid fluctuations in intensity due to tumbling motions, with frequency components up to tens of Hertz. In order to distinguish these fragments and debris from RVs, it is necessary to observe and process data at an appropriately high sampling rate. It is proposed to carry out an initial stage of object discrimination early in the processing chain using data directly from the focal plane array where the individual detector elements can provide the high sampling rate. For the typical sensor parameters listed in Table 2 of Section 2.3.1, the detector sampling period is 120 μsec and the total target observation time (all three colors) is 110 msec. These values will allow measurement of frequency components between 10 Hz and 8.3 kHz. To observe frequency components below 10 Hz, it will be necessary to utilize a slower scan rate, a larger number of TDI stages, and/or a larger color bank offset. If the color bank offset could be increased to 2 degrees (35,000 μrad), for example, the low-frequency limit could be reduced to 1.5 Hz.

2.2.2 Kinetic Kill Debris Rejection. We have carried out a preliminary experiment on the use of velocity filtering for bulk rejection of kinetic kill debris based on motion, using simulation data provided to us by Lincoln Laboratory. Figure 27a-f shows 6 image frames at 5-second intervals. Each frame represents a 120 x 120 pixel wide field-of-view that has been "panned"

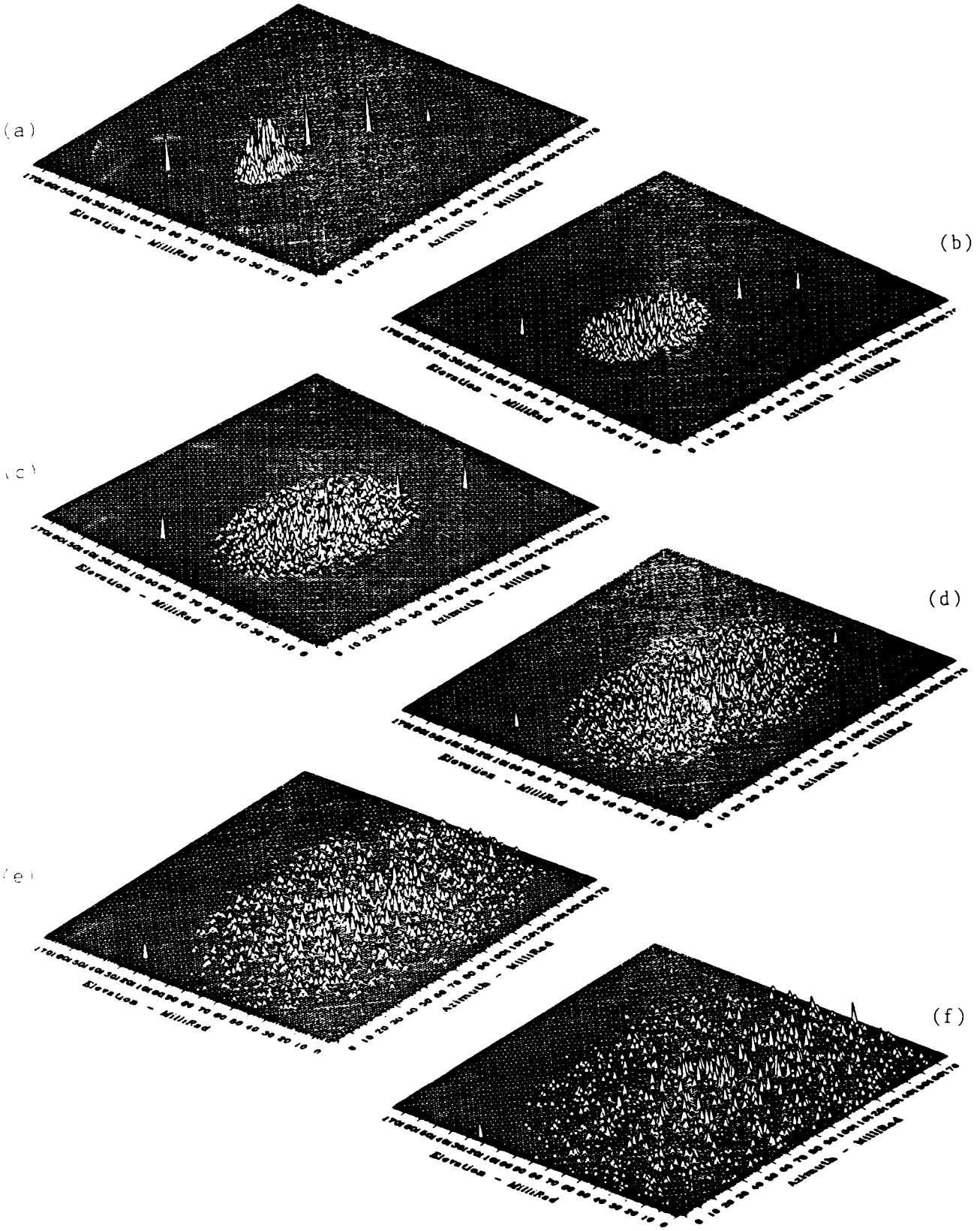

Figure 27 (a-f). Six Images with Four Primary Objects Plus Debris.

to maintain the objects within the frame size shown. There are four main objects representing an RV and its associated decoys plus a debris cloud which expands rapidly to obscure the main objects in the later frames.

Since the main objects would generally be in track prior to the kinetic kill event, we assumed that their known track file velocities could be used to establish the set of four matched filters needed for track continuation. We processed the frame sequence of Figure 27a-f with each of these four velocity filters to obtain the four output frames shown in Figure 28a-d (the output frames are referenced to the time of the sixth input frame in Figure 27f). In each output frame, the intensity of the particular object whose velocity is within the pass band of the filter is enhanced relative to that of the other main objects and the debris. Figure 29 compares the sixth input frame with a composite view of the output frames in Figure 28a-d after thresholding. It is evident that the velocity filters have completely rejected the debris based on motion discriminants alone.

2.3 Processor Implementation

2.3.1 System Configuration. Figure 30 shows a simplified version of the focal-plane array geometry for a passive scanning IR sensor. Each of the L modules contains M rows of detectors, arranged in vertical columns for different wavebands or colors. The columns are separated by a fixed color bank offset. Each of the rows within a given color bank has N detectors for TDI. All of the detectors have the same height and width.

Typical sensor parameters are shown in Table 2.

Table 2. Typical Sensor Parameters

Detector width (height)	50 μrad
Elevation field of view	10^{o}
Azimuth scanning rate	6^{o}/sec
Number of colors	3
Number of TDI stages per color	10
Color bank offset	5000 μrad
Total number of rows	3500
Total number of detectors	105,000

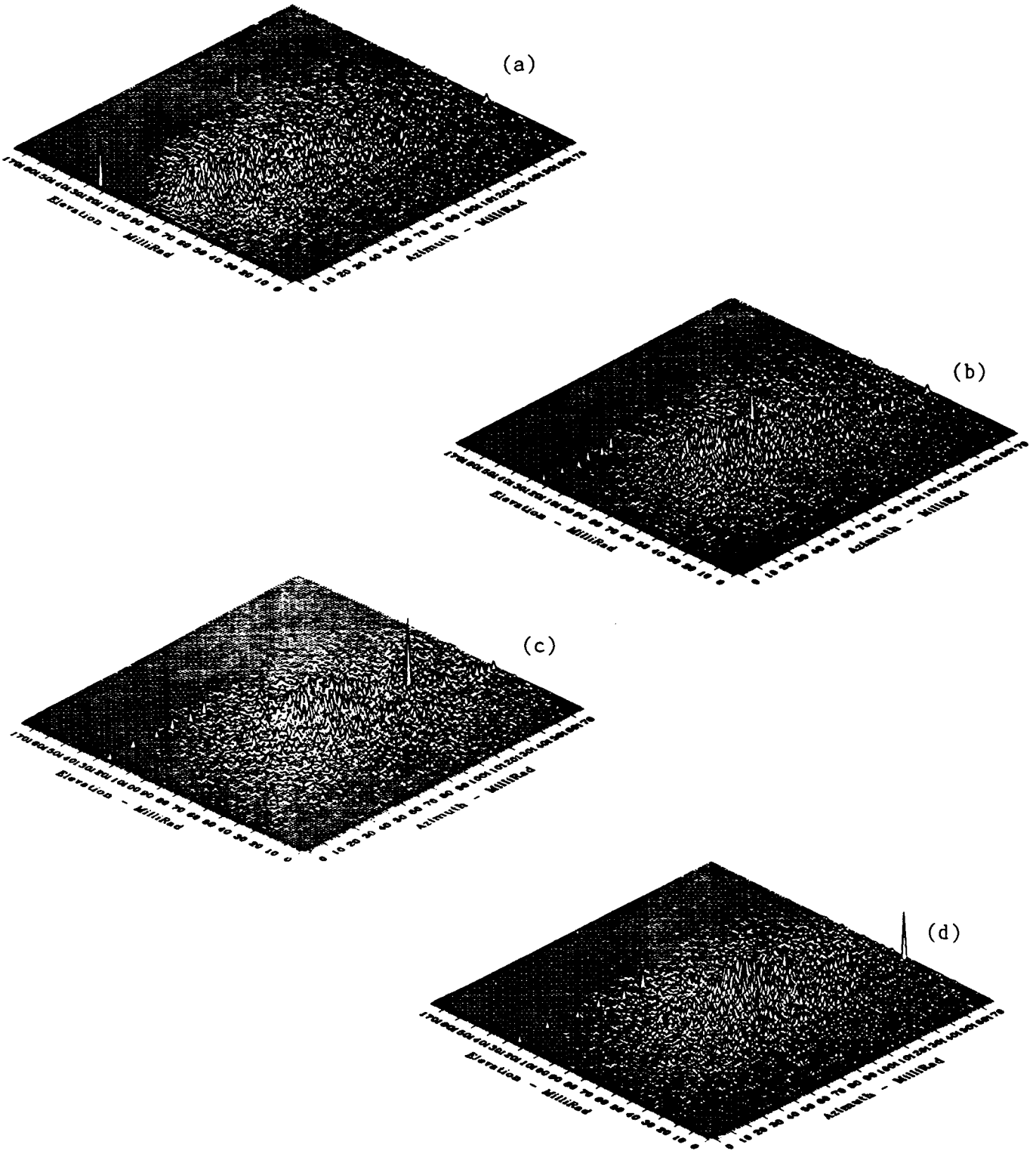

Figure 28 (a-d). Output of Four Velocity Filters.

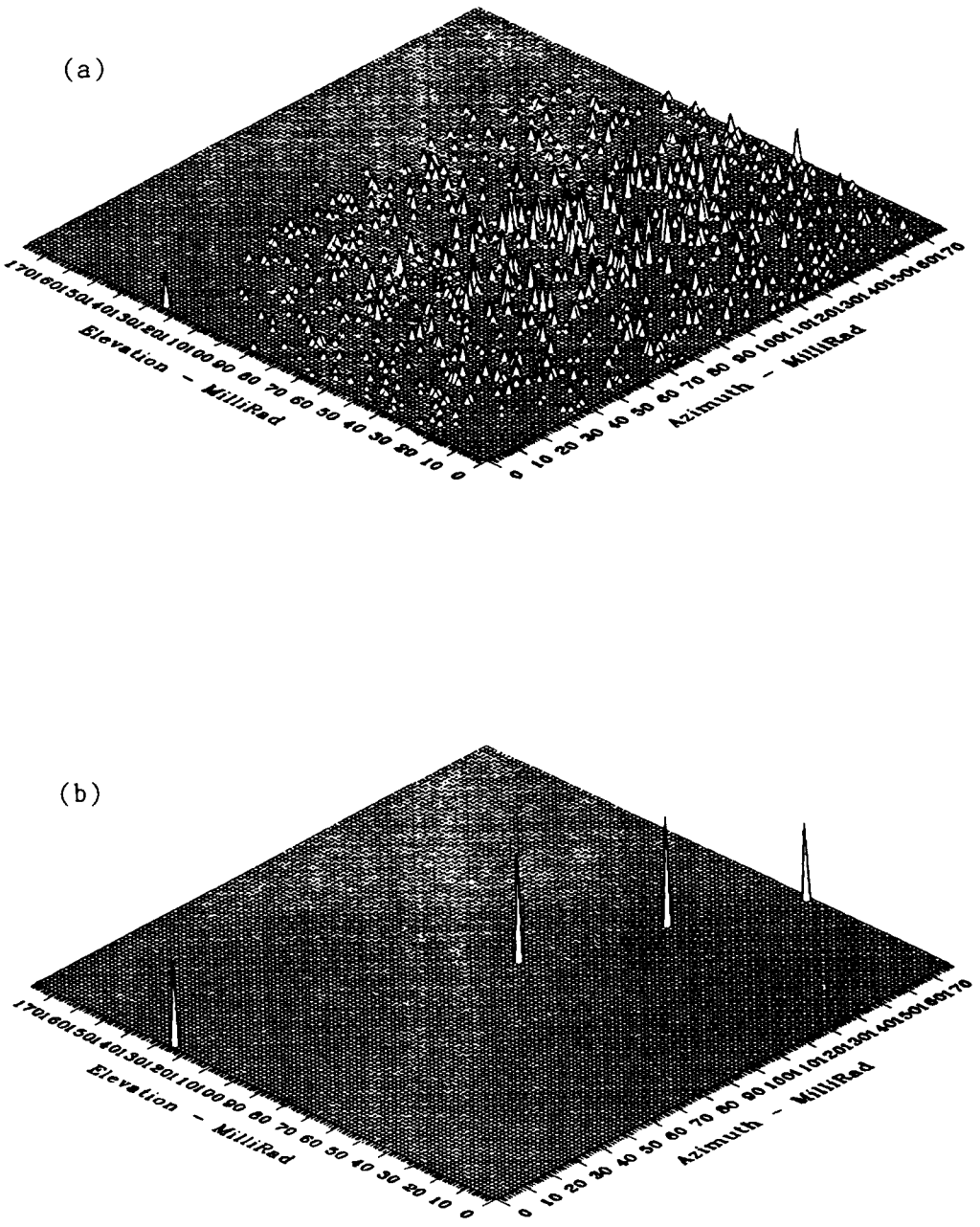

Figure 29 (a,b). Comparison of Sixth Input Frame (a) and Composite Output of Velocity Filters.

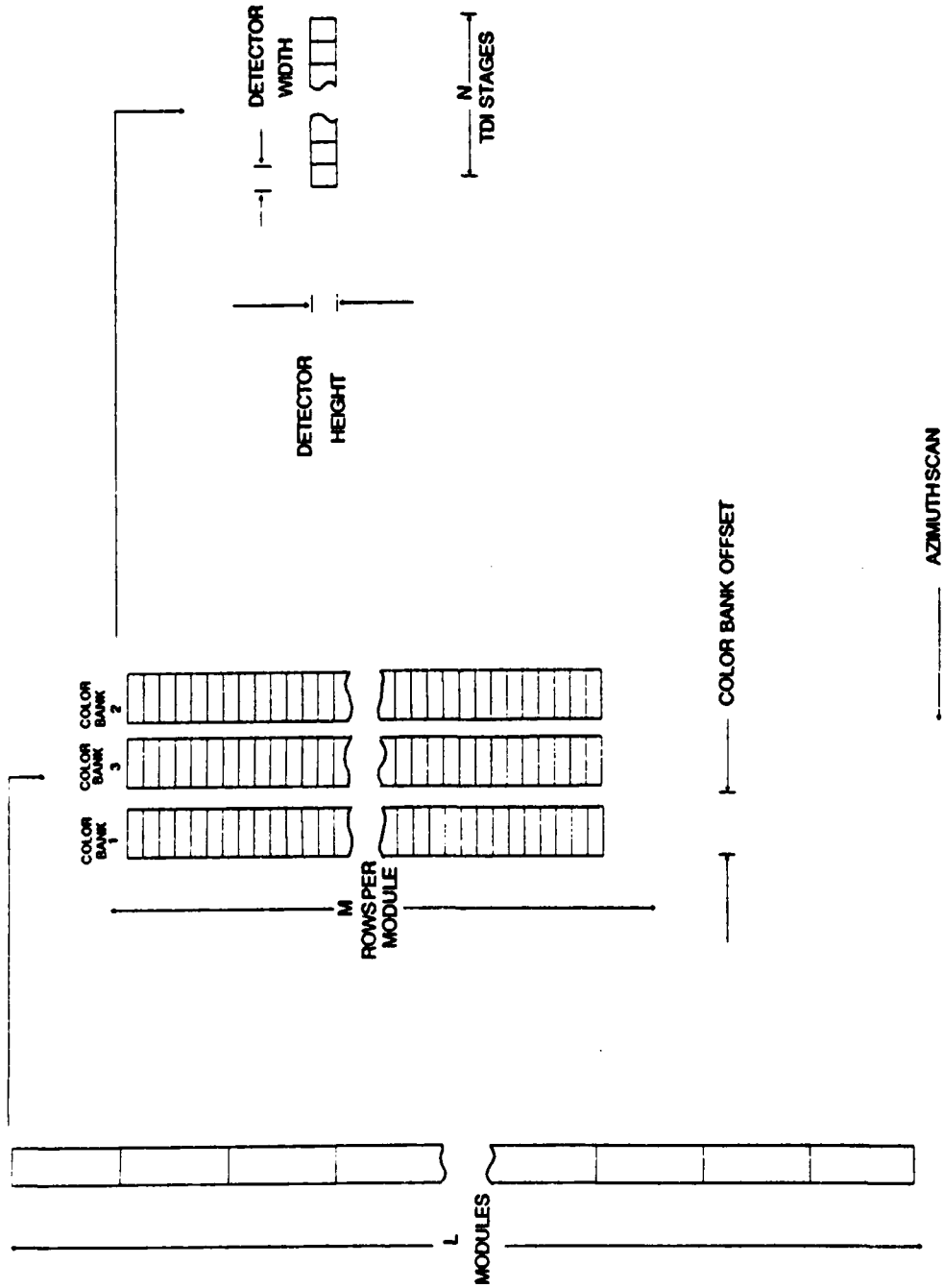

Figure 30. Focal Plane Array Geometry

Detector dwell time	480 μsec
Number of samples per detector dwell time	4
Sampling period per detector	120 μsec
Total detector data rate	0.88×10^9 Hz
Number of modules	64
Module rate	14 MHz
Total target observation time (3 colors)	110 msec

Figure 31 shows the basic system block diagram. The analog pre-processor multiplexes the L sets of module outputs and converts them to digital form (with a typical wordlength of 8 bits). The signal processor has L parallel input channels, each with a data rate of (1/L) times the total detector sampling rate. For the parameters shown in Table 2, there are 64 channels and the data rate per channel is 14 MHz. Each channel provides synchronous, multi-color object intensity data for a subimage with 3500/64 = 55 elevation rows (2750 μrad) as the sensor scans in azimuth. The signal processor executes the "time-dependent" processing functions at an overall data rate equal to the total detector data rate (0.88×10^9 Hz for the sensor parameters in Table 2). The last processing step in the signal processor is a threshold comparison, which reduces the data rate by a large factor. The output of the signal processor is the input of the data processor, which executes the "object-dependent" processing functions. The output of the data processor is a set of messages containing information on the state vectors of the targets for use by other sensors, by the battle management system, or by weapons.

2.3.2 Basic Processor Requirements. The basic processor requirements are throughput, programmability, reliability, radiation hardness, size/weight/power, and architectural invariance. These are discussed briefly below.

a. Throughput. The required throughput for time-dependent processing is on the order of 10^{11} operations per second. For object-dependent processing, the throughput required is a function of the number of threat objects as well as the algorithms used for scan-to-scan correlation. In all cases, the required throughput is beyond the capability of a single processor, and some form of parallel processing is essential.

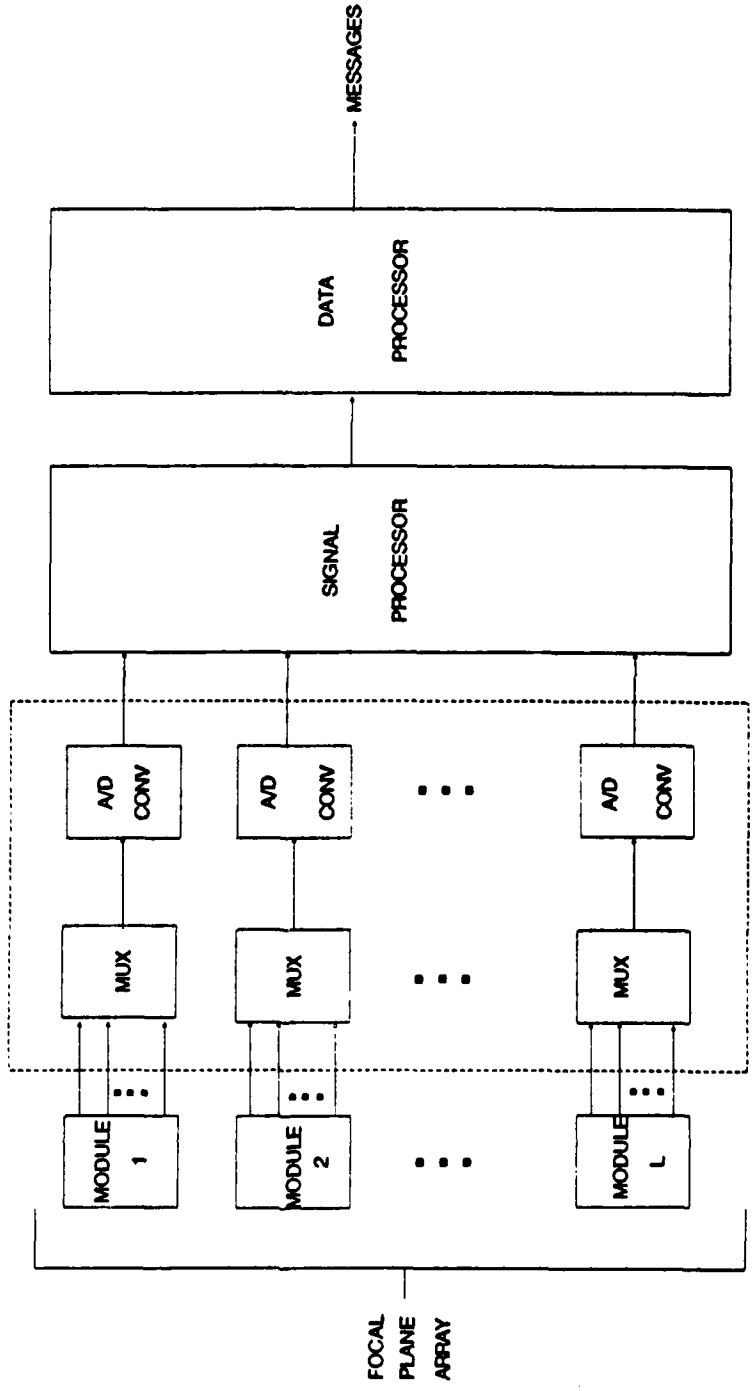

Figure 31. Basic System Block Diagram.

b. Programmability. In general, a high degree of programmability is desired. However, there are significant tradeoffs between programmability, throughput and hardware complexity. Hardwired processors, which have a low degree of programmability, typically have an order-of-magnitude throughput advantage over general-purpose programmable processors for the same size, weight and power.

c. Reliability. Spaceborne processors require very high long-term reliability, typically 0.9 or better for 10 years. The only way in which such a high reliability can be obtained with current technology is through the use of redundancy and fault tolerance techniques.

d. Radiation hardness. All spaceborne processor hardware must be capable of being hardened to withstand both natural radiation and nuclear weapon effects. Typical levels required are as follows:

Total dose	2×10^5 rad(Si)
Dose rate:	
Upset	10^{10} rad(Si)/sec
Survival	10^{12} rad(Si)/sec
Neutron fluence	10^{13} N/cm^2
Single-event upset	10^{-10} errors/bit/day

e. Size/weight/power. These parameters, especially power, must be as small as possible.

f. Architectural invariance. It is highly desirable that the processor be designed in such a way (1) that improved hardware technology (e.g., smaller IC feature size) can be utilized with minimum design modifications as it becomes available and (2) that it can be extended for higher performance by the simple addition of more processing elements.

2.3.3 Alternative Approaches. There are two basic approaches to processor design. One approach is to use special-purpose hardwired circuitry, characterized by low programmability but high speed. This approach is commonly employed for the signal processor because of its high throughput requirements and because the algorithms utilized for time-dependent processing are considered to be relatively simple and unlikely to require changes. This processing is most naturally carried out by identical parallel processors, one per channel.

The second basic approach to processor design is to employ a general-purpose, programmable parallel processor. This approach is commonly employed for the data processor, but it can also be applied to the signal processor (especially if augmented by special-purpose co-processors for execution of algorithms requiring exceptionally high throughput).

A wide variety of parallel processor architectures are possible for the general-purpose approach. At one extreme, "coarse-grained" machines like the Cray X-MP use a small number of powerful processors. Each processor uses a sophisticated pipelined architecture and is built using the fastest available circuit technology. This approach is not suitable for spaceborne processors because of the large number of devices, the large size and weight, and the high power required.

At the opposite extreme are extremely simple, bit-serial processors. Although any one of these processors is of little value by itself, the aggregate computing power can be very large when many are coupled together. A well-known example of this approach is the Thinking Machines Corporation Connection Machine, which uses tens of thousands of one-bit processing elements to achieve throughputs in excess of 2500 MIPS. Although the Connection Machine itself is not suitable for space applications because of its large size, weight and power consumption, it is possible to design other fine-grained machines which are. One such machine is the 3-D computer under development at Hughes Aircraft Company, which employs wafer-scale integration. Others are (1) the DAP 510 Array Processor from Active Memory Technology, Inc. and (2) the Associative String Processor (ASP) developed by Brunel University and Aspex Microsystems Ltd., which will be described shortly.

The third, intermediate approach combines an intermediate number (tens to hundreds) of medium-grain microprocessors. Among the many possible architectures in this category is the so-called multicomputer network. Each node in the network contains a processor with a locally addressable memory, a communications controller capable of routing messages, and a small number of connections to other nodes. In general, there is no shared memory; the cooperating tasks of a parallel algorithm must execute asynchronously on different nodes and be communicated solely by message passing. Examples include the Caltech Cosmic Cube and its descendants (Intel iPSC, Ametek System/14, Ncube/ten, FPS T Series, JPL Mark-III), the Advanced Onboard Signal

Processor (AOSP), and a variety of other designs based upon MIL-STD-1750A or RISC processors combined with vector processors.

The multicomputer network can, at least in principle, be implemented with existing ICs or extensions of existing ICs. Also, it can be programmed in extensions of familiar serial languages. There are three major problems, however, with the multicomputer network. The first problem is its need for sophisticated software structuring to provide (1) load balancing of the computations at the different nodes and (2) limiting of inter-node communications rates to available bandwidths. The second problem is the difficulty of replacing faulty nodes with spare nodes to achieve fault tolerance, due to the complexity of the communications network. The third problem is the limited throughput per unit of hardware, due to the relative complexity and non-homogeneous nature of the circuitry.

2.3.4 New Algorithms. Processor requirements for the new algorithms discussed in previous sections differ from those for traditional algorithms. In the case of the signal processor, the addition of first-stage object discrimination to the time-dependent processing functions makes a high degree of programmability imperative. This is because the most effective algorithms and parameters to be employed for first-stage object discrimination are not known at this time and in fact will be subject to change as experimental data and the results of analyses and simulations become available over the coming years. Therefore a general-purpose, programmable processor should be used each of the parallel channels rather than a special-purpose, hardwired processor.

In the case of the data processor, the addition of velocity filtering than traditional multi-target track association algorithms leads to special requirements. The basic shift-and-add velocity filter algorithm has a very high degree of parallelism and is extremely well suited to execution on a fine-grained machine with a large number of processing elements that can provide the high throughput required. To illustrate, we will consider the "hit list" form of the algorithm described earlier. Assume that there are H hits per frame and V velocity filters, i.e., an average of H/V hits per velocity filter. Assume also that each list entry

contains two bytes for azimuth and elevation tags plus one byte for intensity data. For a pair of frames, the first operation is to offset the azimuth and elevation tags in the first frame by constants corresponding to the desired 2-D shift. This requires 2H one-byte additions. The next operation is to find entries in the list for the second frame which match first-frame entries in both azimuth and elevation. This requires H^2 two-byte comparisons. The third operation is to add the intensities for the matching entries; this requires H/V one-byte additions. The foregoing process is then repeated every frame period for the remainder of the V velocity filters. Since H is typically much larger than H/V, the comparisons will dominate the computation, so that the total number of operations (two bytes each) required per frame period is approximately VH^2. For typical values H = 120,000, V = 3000 and a frame period of 10 seconds, the required throughput is 4.3×10^{13} operations per second. For an image sequence of four frames with staggered processing over two scan periods, the throughput needed is 2.2×10^{13} operations per second.

It is clear from the above example that the velocity filter processor should be massively parallel, with the ability to perform parallel comparisons as rapidly as possible. It should also be able to perform 2-D shifting of list entries by parallel addition, and should have a high I/O bandwidth. The local memory of each PE should be able to store at least one list entry (three bytes), and the number of processing elements (PEs) should be at least equal to the number of hits per frame in order that all of the list entries can be stored within the local memories of the PEs.

The above requirements can be met by either a special-purpose or general-purpose design. In either case, the machine will be fine-grained and massively-parallel, with a large number of PEs, a small amount of local memory per PE, and an efficient method of performing comparisons between data stored in PEs at arbitrary locations. The buffer memory required will be relatively modest in size; for the foregoing example, only 1.8 MBytes are required (five frames of single-color hit data).

2.3.5 Associative String Processor (ASP). Of the various fine-grained parallel processor architectures we investigated for execution of the new algorithms, including both special-and general-purpose types, the Associative String Processor appears to be the most attractive. The basic ASP concept has been developed by Brunel University and Aspex Microsystems Ltd. in the U.K.

Early research dealt with content-addressable memories and microprogrammed control logic, and included the construction of a prototype machine and the fabrication of custom LSI chips. A VLSI implementation program initiated in 1981 has involved the development of a cell-based design methodology for full-custom VLSI, ULSI and WSI devices, followed by the design and fabrication of several chips using two-micron, double-metal CMOS technology. These include a device with 64 processing elements intended for construction of ASP demonstration systems. This device is currently being fabricated by Plessey. The program has also involved the development of software simulators and the generation of microcode for machine instructions, as well as extensive system simulations and application programming studies.

Current investigations at Brunel and Aspex include the development of WSI (wafer-scale integration) and ULSI versions of the Associative String Processor, with funding provided by both the U.K. and U.S. governments.

The ASP architecture provides a homogeneous, reconfigurable, programmable, fine-grained parallel processing structure particularly well suited to implementation with state-of-the-art microelectronics technology. ASP machines are general-purpose in nature and possess the flexibility for use over a wide range of applications. However, they can also be optimized for certain classes of work such as image processing with accompanying increases in performance and reductions in complexity.

We now discuss the key features of the ASP architecture. First, it employs associative processing techniques. Such techniques permit extremely efficient, high-speed communications and control in fine-grained parallel machines through the use of content-matching rather than location-addressing. Content-matching utilizes broadcasting of data, local comparison, binary response and key-tagging. Since the need to transfer data between processing elements is drastically reduced, a simple on-chip inter-APE communication network can be employed without significant loss of performance. Furthermore, the lack of location-addressing greatly simplifies the incorporation of redundancy and reconfiguration for fault tolerance.

The second feature is the use of a string, or linear array topology. This is the simplest possible structure for VLSI implementation as well as for data communications. No loss of generality results from the selection of this topology since all abstract data structures such as sets, text strings, n-dimensional arrays, tables, trees, directed graphs, etc. can be mapped to

string format. Furthermore, in conjunction with the use of associative processing and modular VLSI implementation, it permits essentially unlimited extensibility: the ability to add as many processing elements as desired without system redesign.

Figure 32 shows an ASP string of identical APEs (Associative Processing Elements) with its controller. Each APE is connected to an Inter-APE Communications Network, which runs in parallel with the APE string. All APEs share common Data, Activity and Control buses and a single Match Reply feedback line as shown.

In operation, an ASP string supports a form of set processing in which the subset of active APEs (i.e., those which match broadcast data and activity values) support scalar-vector and vector-vector operations. Matching APEs are either directly activated or act via inter-APE communications to indirectly activate other APEs. The Match Reply line indicates whether some or none of the APEs match.

The contents of an APE, shown in Figure 33, comprise a Data Register, an Activity Register, a Comparator and logic for local processing and communications with other APEs.

The ASP Controller executes a sequence of ASP procedures, stored as microprograms and called by an application program in a host machine. The Controller also handles the buffering of input and output data.

It should be noted that inter-APE communications do not entail the transfer of actual data. Instead, and much more simply, communication is restricted to the high-speed transfer of activity signals (one or several bits each) between neighboring or selected remote APEs (viz., those matching the selection criteria). Since APEs can be easily activated by such content-addressing and and their data content processed in situ, the time-consuming movement of data is reduced to an absolute minimum.

The ASP has an extremely high performance-to-cost ratio relative to other fine-grained parallel processors, primarily because its architecture is closely matched to both the capabilities and constraints of state-of-the-art VLSI technology. For example, the ASP topology provides a high APE packing density, with extensive cell replication and simple cell interfacing, as well as a highly compact inter-APE communications network.

The current VLSI ASP chip has been specifically developed by Apex for building demonstration ASP systems, and contains 64 APES.

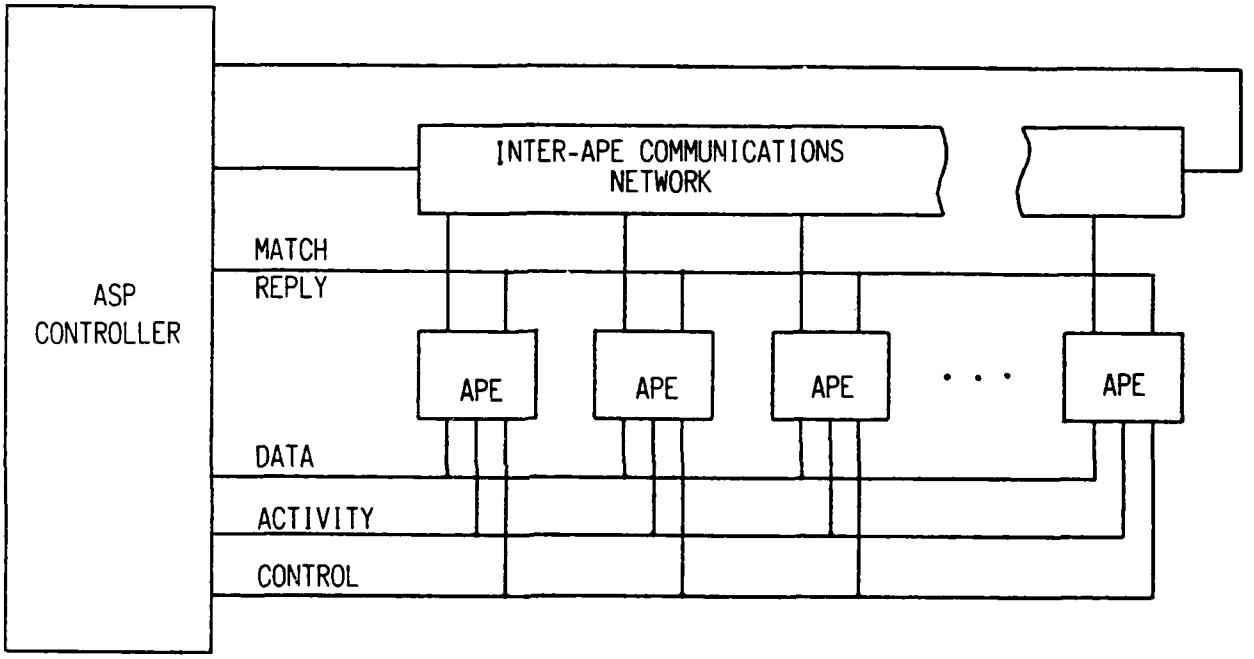

Figure 32. Block Diagram of an Associative String Processor (ASP).

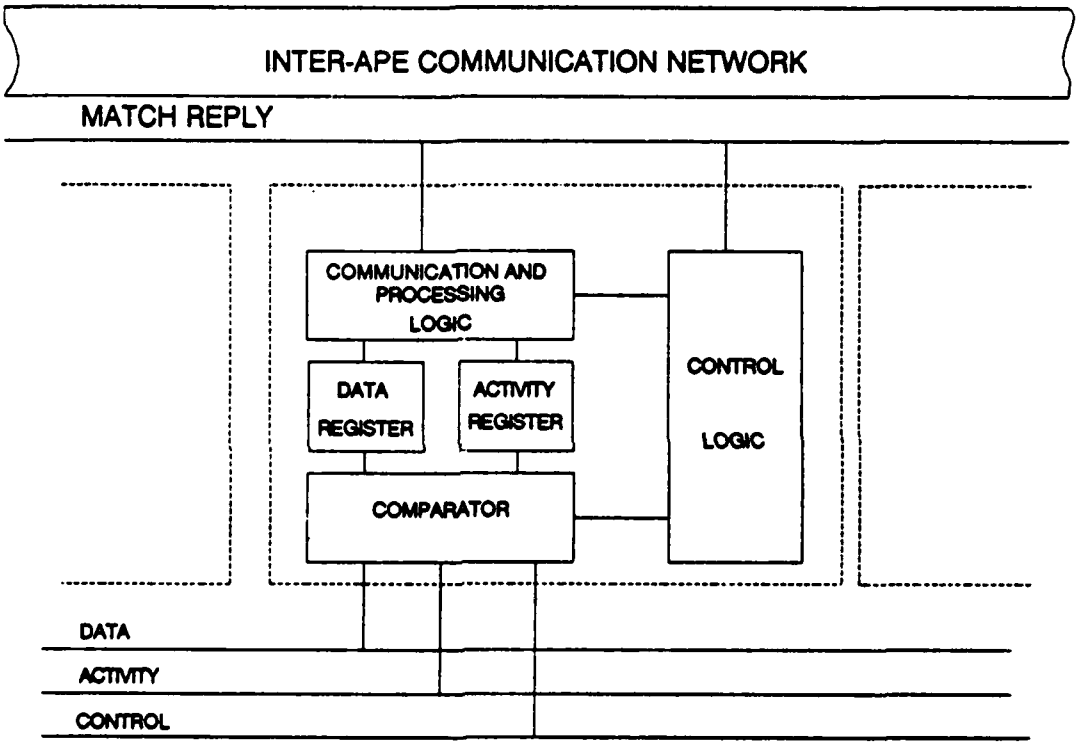

Figure 33. Associative Processing Element.

Each APE contains 37 bits of associative local store, a bit-serial full adder, two single-bit latches and a communications channel port. The chip contains, in addition to the 64 APEs, a complete inter-APE communications network with input and output ports plus data, activity, control and match reply buses. The clock rate is 20 MHz, the power consumption is 320 milliwatts (peak), and the number of I/O pins is 84.

Large numbers of APEs (tens or hundreds of thousands) can be provided by interconnecting multiple ASP chips. The use of hybrid wafer-scale integration, with multilayer thin-film substrate interconnections, permits extremely compact packaging (over 100K APEs in a four-inch cube, for example). Compact packaging can also be achieved by the use of monolithic WSI. By adding fault-tolerant cells, ASP structures comprising tens of thousands of APEs can be integrated on a single undiced silicon wafer. Work in this area is currently underway at Brunel and Aspex, including the fabrication and evaluation of WSI test chips and working WSI device demonstrators. Future WSI devices will each contain 8K APES with 75 nsec cycle time and a power consumption of 10 watts.

It should be noted that the ASP architecture is particularly well suited to the incorporation of fault tolerance for high long-term reliability, owing to its construction from a large number of identical APEs, to simple APE inter-connection, and to lack of location-dependent addressing. For space applications, a pool of spare unpowered APEs can be provided. Upon detection of a faulty APE in the operating system, the faulty APE is bypassed and a spare APE is connected by applying power. The fact that the spare APEs are normally unpowered increases their reliability (by a factor of approximately ten) and limits system power consumption to operating APEs only.

2.3.6 Signal Processor Implementation. The signal processor which executes first-stage object discrimination algorithms and the time-dependent processing functions discussed in Section 2.3.1 can be implemented very efficiently with the ASP. Since the ASP is a general-purpose, programmable machine, it provides the flexibility needed for evolutionary changes in the object discrimination algorithms. The number of APEs and WSI devices required for the typical system parameters given in Table 2 is determined roughly as follows. Based upon rough estimates and experience with other ASP applications, we assume that 250 APE cycles per input data sample are required. The throughput per channel is then 250 times the module rate of 14 MHz, or 3.5×10^9 APE

cycles per second. Since the APE cycle time is 75 nsec, the number of APEs required per channel is 270. Since there are 64 channels, the total number of APEs needed is 64 x 270 = 17,300. With 8K APEs per WSI device, a total of 3 WSI devices are required for the APE strings. The total power consumption will be approximately 30 watts.

2.3.7 Data Processor Implementation. We now consider implementation of the data processor with ASP architecture and chips. We will limit our discussion to execution of the stressing computations, which are those for the velocity filter algorithms. We assume, in addition to the typical sensor parameters given in Table 2, the following:

a. Frame size = 10^{o} x 60^{o};
b. Frame period = 10 sec;
c. Number of frames in image sequence = 4;
d. Number of hits per frame = H = 120K (corresponds to 3000 RVs, 40 decoys per RV);
e. Number of velocity filters = V = H/40 = 3K (corresponds to number of velocity clusters).

As before, it is assumed that the APE cycle period is 100 nsec and that each APE has enough local memory to store one hit list entry. For simplicity, we limit the discussion to single-color data.

The basic processing structure consists of a buffer memory capable of storing the hit lists for five frames (1.8 MBytes) plus an ASP having N parallel APE strings. Each string has H APEs so that it can store data for one frame within the its APE local memories. Loading takes place via the data bus, one APE at a time.

To implement a given velocity filter within a string, we carry out the following steps:

-Step 1: Load frame 1 and shift via addition of constants to the azimuth and elevation tags. This step requires H + 16 machine cycles.

-Step 2: Load frame 2 and compare with the shifted version of frame 1. Add the intensities for the matching entries together and shift by the same amount as in step 1. These operations require approximately 3 H + 16 cycles.

-Step 3: Repeat step 2 using frames 3 and 4. These steps require approximately 6 H + 32 cycles.

The above steps require a total of approximately 10 H cycles.

To implement H/40 velocity filters, we repeat the above steps using multiple frame loadings from the buffer memory. In order to complete all of the computations within two frame periods (using staggered frame sequences), the number of strings required is

$$N = 10\ H \times 100\ \text{nsec} \times (H/40)/20\ \text{sec}$$
$$\approx 1.25 \times 10^{-9}\ H^2$$

The total number of APEs required is then

$$H \times N = 1.25 \times 10^{-9}\ H^3$$

The number of APEs required can be greatly reduced by partitioning the frames and adjusting the string lengths in accordance with the number of hits per subframe (which can be determined by counting the entries during loading of the buffer memory). Let S be the number of subframes per frame and H_i be the number of hits in the i-th subframe. The summation of H_i from i = 1,...,S equals the total number of hits H. If the H_i are small enough so that only one string of H_i APEs is sufficient to compute all of the velocity filters for the i-th subframe, then the total number of APEs required is equal to H. This condition is fulfilled if

$$H_i \leq 2.83 \times 10^4$$

For example, a uniform hit distribution with S = 5 gives H_i = 24 K and requires a total of 120 K APEs. The same number of APEs will handle a non-uniform distribution of hits in which all of the hits are located within five of the subframes (for any value of S equal to five or more).

It was implicitly assumed in the above that the targets do not cross subframe boundaries during processing of the frame sequence. In order to insure that this does not occur, the subframes should be overlapped. As an illustration, we analyzed midcourse data from a simulation of the 1995 Centerline MEA-1 Threat. Within a typical 3180-target subset, the maximum movement per target during a 40-second interval was 0.9° in azimuth and 0.03° in elevation. With the system parameters assumed and S = 5, the subframe size is $2^\circ \times 12^\circ$. The amount of overlap required in this case is obviously very small.

The number of APEs required to accommodate a range of threat characteristics and system geometries is scenario-dependent and requires detailed analysis. However, it is believed that several hundred thousand APEs will suffice for most or all scenarios of practical importance. With 8K APEs per

WSI device, a total of 32 devices will provide 256K APEs. The estimated power consumption at 10 watts per device will be 320 watts.

It should be noted that the ASP implementation of the new algorithms for the data processor will provide much higher performance than that of traditional approaches alone. It will handle much larger numbers of threat objects under simultaneous launch conditions, and it will be much more robust with respect to merging/crossing tracks, temporary loss of sensor data, extended bright backgrounds, and other abnormal conditions. Other advantages include ease of incorporation of fault tolerance, feasibility of extremely compact packaging, and ease of translation of ICs to CMOS/SOS or other radiation-hard form.

3. FINDINGS OR RESULTS

Results of the Phase I research program are summarized below.

Velocity filter approach to target detection and tracking.

a. Theoretical studies. These included the derivation of the optimum (maximum-likelihood) filter for detecting a target moving through a sequence of N passive sensor image frames with a known, assumed or estimated vector velocity. The target signal and noise observations in the image pixels were modeled as independent Poisson processes to account for the statistical nature of the photodetection process. For the case of a uniform noise background (due to either homogeneous infrared clutter or detector noise), it was shown that the optimum detection algorithm for N consecutive image frames could be implemented as follows: (1) subtract the mean noise level from every pixel of each image frame, (2) shift-and-add the N frames according to the target velocity measured in pixels per frame, (3) correlate the combined frame with the spatial target signal, (4) threshold the result. These basic results were extended to scenes with spatially non-homogeneous random backgrounds due to the presence of strong IR point sources (such as stars) or extended clutter regions (e.g., clouds) in the sensor field-of-view. Expressions for the overall detection performance were also derived.

b. Experimental studies. Algorithm simulation experiments were conducted on real and simulated passive sensor imagery to validate some of the basic aspects of velocity filter detection and tracking performance. A variety of target and background scenes were processed, including a subset of the 1995 Centerline MEA-1 Threat from TRW, a 112-target set used for scan-to-scan correlation studies at the MIT LIncoln Laboratory, and several sets of digital imagery recorded by the Lincoln Laboratory 30-inch "Quad-Camera" optical telescope in New Mexico. The latter source of imagery was of particular interest since it was collected on an actual CCD focal plane array. The Quad-Camera datasets we processed included a sequence of scenes of a fast-moving satellite; these were used for testing the optimum velocity filter. Scenes of the Great Nebula in the constellation Orion were also used to study background suppression. All of these scenes exhibited the

effects of CCD photodetector noise in addition to a number of other artifacts including image jitter, clipping due to limited dynamic range, and "holes" caused by dead detector cells.

Object discrimination.

a. First stage discrimination. Preliminary studies were carried out on the possibility of performing an initial stage of object discrimination using data directly from the focal plane array where the individual detector elements can provide a high sampling rate. A high rate is required to observe and process rapid fluctuations in intensity which are believed to be characteristic of booster fragments and kinetic kill debris due to tumbling motions.

b. Kinetic kill debris rejection. We performed a preliminary experiment on the use of velocity filtering for bulk rejection of kinetic kill debris based on motion, using simulation data provided to us by Lincoln Laboratory. The experiment utilized six image frames at five-second intervals. Each frame has four main objects representing an RV and its associated decoys plus a debris cloud which expands rapidly to obscure the main objects in the later frames. Since the main objects would generally be in track prior to the kinetic kill event, we assumed that their known track file velocities could be used to establish the set of four matched filters needed for track continuation. The experiment was successful in that the velocity filters completely rejected the debris based on motion discriminants alone.

Processor implementation.

a. Processor requirements and alternative architectures. We investigated a wide variety of parallel architectures, both special- and general-purpose in nature, including coarse-grained, medium-grained and fine-grained machines. We also derived processor requirements for execution of the new algorithms associated with first-stage discrimination and velocity filtering. These algorithms encompass both the time-dependent and object-dependent portions of the overall processing chain and are executed in the signal processor and data processor, respectively. In the case of the signal processor, we concluded that a high degree of programmability was needed because the most effective algorithms and parameters to be employed for first-stage object discrimination are not known at this time and in fact

will be subject to change as experimental data and the results of analyses and simulations become available over the coming years. In the case of the data processor, we concluded that the architecture should be fine-grained and massively-parallel, with a large number of processing elements, a small amount of local memory per processing element, and an efficient method of performing comparisons between data stored in processing elements at arbitrary locations.

b. Associative String Processor (ASP). Of the various fine-grained parallel processor architectures we investigated, the Associative String Processor under development in the U.K. at Brunel University and Aspex Microsystems Ltd. appears to be the most attractive. This machine employs content-matching rather than location-addressing, drastically reducing the need for data communications and simplifying the incorporation of fault tolerance. It also has essentially unlimited extensibility and an extremely high performance-to-cost ratio resulting from the close match between machine architecture and VLSI technology.

4. POTENTIAL APPLICATIONS OF THE EFFORT

The new approaches to object acquisition, tracking and discrimination via bulk processing investigated during this project should be applicable to a wide range of strategic defense sensors, including BSTS, SSTS, AOS and GBSS as well as to fire control systems for kinetic and directed energy weapons. In particular, it should be possible to augment classical track-association techniques with velocity filter algorithms to obtain higher, more robust performance as well as reduced processor hardware requirements.

Other potential applications include military reconnaissance, surveillance and intelligence systems, including those for cruise-missile detection, as well as commercial robotics and computer vision systems.